WebSphere® Adapters

IBM

Version 6.0.2

Adapter for JDBC User Guide

Note

Before using this information, be sure to read the general information in “Notices” on page 165.

22December2006

This edition applies to version 6, release 0, modification 2 of WebSphere Adapter for JDBC (product number 5724-L77) and to all subsequent releases and modifications until otherwise indicated in new editions.

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

Contents

Chapter 1. About this information

This documentation is for integration developers who implement, configure, and deploy WebSphere® Adapter for JDBC. To use it, you should understand business integration concepts and possess certain technical skills.

Integration developers design, assemble, test, and deploy business integration solutions. This information is for those who are deploying WebSphere Adapter for JDBC in a solution that requires data exchange between enterprise information systems (EIS) and Java™ 2 Platform, Enterprise Edition (J2EE) applications. To use it, you should understand and have experience with the following concepts, standards, and tools:

- The business solution and environment.
- Databases, data access issues, transactional models, and connections across heterogeneous relational databases, queues, and Web services.
- Business integration mechanisms, including the Service Component Architecture (SCA) programming model and the Service Data Objects (SDO) data model.
- The J2EE standard and J2EE applications.
- The capabilities and requirements of WebSphere Process Server or WebSphere Enterprise Service Bus, depending on the host used in the environment. You should know how to configure and administer the host server and how to use the administrative console.
- The tools and capabilities provided by WebSphere Integration Developer. You should know how to use these tools to wire components and complete other integration tasks.

To complete the deployment, you should know how to perform the following tasks:

- Create required scripts, tools, and templates for both testing and deployment
- Resolve interdependencies between entities such as enterprise beans, workflows, and Web pages
- Write procedures to use database access logic efficiently
- Build data models for external data access tools
- Implement security measures

Chapter 2. What's new

WebSphere Adapter for JDBC, version 6.0.2 provides enhancements to version 6.0 of the adapter.

New in this release

WebSphere Adapter for JDBC, version 6.0.2, includes a number of enhancements; for example, the adapter supports query business objects, custom event processing, and stored functions.

New in version 6.0.2:

- Support for the SQL data types CLOB and BLOB.
- Support for stored functions. The adapter checks for ReturnValue application-specific information and, if it exists, the adapter runs the stored function.
- Custom event processing: Custom queries use either standard SQL, a stored procedure, or a stored function.
- EDT is no longer used for assured once delivery. A new Activation specification property, AssuredOnceDelivery, is provided for this function.
- A new filter can be used to narrow the list of stored procedures in enterprise metadata discovery when associating a stored procedure with a business object
- The adapter can filter events to be processed by business object type using the Activation specification property EventFilterType, and it can filter events by timestamp using the property FilterFutureEvents.
- The ability to generate query business objects from user-specified select statements during enterprise service discovery.
- The Execute operation to support stored procedures and stored functions.
- Support for establishing database connections through the DataSource instance. A DataSourceJNDIName property has been added to the Managed (J2C) connection factory and Activation specification properties.
- High availability support for inbound processing. For more information, see ″WebSphere Adapters in clustered environments.″

Release notes

The release notes for WebSphere Adapter for JDBC, version 6.0.2 summarize new features and functions in this release and document any known workarounds.

Release notes for this adapter can be found at the following Web site: Adapter for JDBC release notes.

Chapter 3. Introduction to WebSphere Adapters

IBM® WebSphere Adapters make it possible for Java 2 Platform, Enterprise Edition (J2EE) components, such as new e-business applications, to communicate with resources on an enterprise information system (EIS). An EIS is the information infrastructure for an enterprise (for example, an enterprise resource planning [ERP] system).

A WebSphere adapter acts as an intermediary between the J2EE component and the EIS, so that the J2EE component does not need to understand the low-level API or data structures of the EIS.

WebSphere Adapters can be one of two types: application or technology.

- Application adapters connect to existing packaged applications (such as SAP Software, Siebel, PeopleSoft Enterprise, and JD Edwards EnterpriseOne) so that you can make use of data and services specific to the applications.
- Technology adapters provide connectivity to data through such technologies and protocols as relational databases, flat files, e-mail messages, and FTP.

As part of the WebSphere family of products, WebSphere Adapters work with WebSphere Integration Developer and either WebSphere Process Server or WebSphere Enterprise Service Bus.

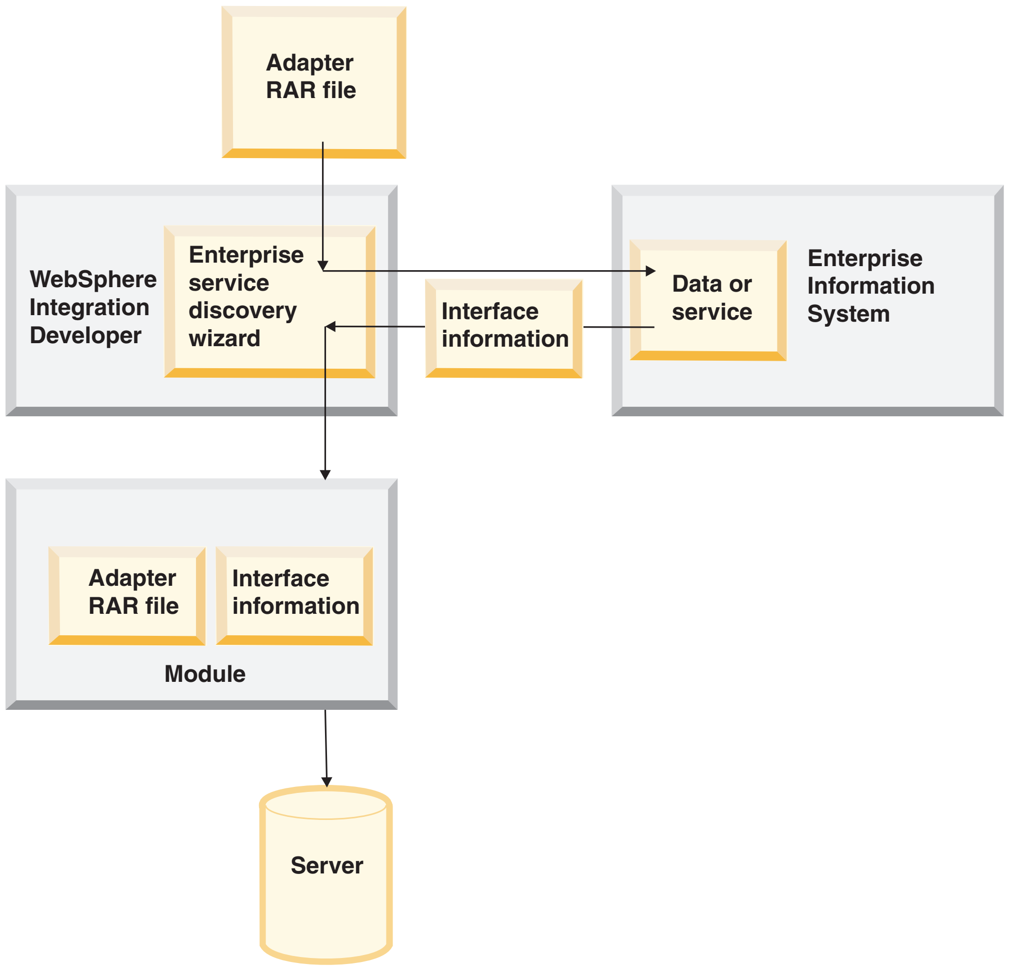
- WebSphere Integration Developer is the tooling environment for the WebSphere adapters.

  You use WebSphere Integration Developer to assemble a module that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus. From within WebSphere Integration Developer, you import the adapter (which is packaged as a resource adapter [RAR] file) and connect to the EIS. The enterprise service discovery wizard of WebSphere Integration Developer looks for data and services on the EIS and creates the interface information needed to gain access to the data and services. Finally, WebSphere Integration Developer generates a module that includes the adapter and the interface information.
- WebSphere Process Server or WebSphere Enterprise Service Bus is the runtime environment for the WebSphere adapters.

  You deploy the module generated by WebSphere Integration Developer to one of the servers.

The generation and deployment of the module are illustrated in the following figure.

Figure 1. How a module is generated and deployed


Adapter RAR file
WebSphere Integration Developer
Enterprise service discovery wizard
Interface information
Data or service
Enterprise Information System
Adapter RAR file
Interface information
Module
Server

Chapter 4. Introduction to WebSphere Adapter for JDBC

WebSphere Adapter for JDBC is a resource adapter that provides bidirectional connectivity between J2EE applications and enterprise information systems (EISs). For such applications, the exchange of data, which is in the form of business objects, happens at the database level.

The Adapter for JDBC communicates with an EIS which is a database provider. Updates to a database might need to be applied to another enterprise information system, and changes to data in an EIS might need to be applied to a database. The resource adapter can integrate with any application built on a database having a JDBC driver that supports the JDBC 2.0 or later specification. Examples of such databases include IBM DB2®, Oracle, Microsoft® SQLServer, Sybase, and Informix®.

To support integration, the resource adapter processes requests received from any EIS and processes events generated as a result of database updates. The adapter transmits these events to various predefined endpoints in the application server. *Endpoints* are J2EE applications or other client consumers of the event. Data updates that are made to tables in a database can be automatically propagated to other applications connected to the application server, such as Siebel, PeopleSoft, and Oracle applications, through event notifications posted in an event store. The adapter updates the database tables using SQL queries or stored procedures, as specified in the business objects. *Business objects* are containers for application data that represent business functions or elements, such as a customer or an invoice.

Hardware and software requirements

Before installing Adapter for JDBC, you must verify that your environment meets the necessary requirements. These requirements fall into two categories: supported platforms for running the adapter installer, and hardware and software requirements for configuring, deploying, and running the adapter.

Supported platforms for running the adapter installer

The supported platforms for running the adapter installer are located in the ″Installing″ section of Installing IBM WebSphere Adapters.

Hardware and software requirements for configuring, deploying, and running the adapter

The hardware and software requirements for configuring, deploying, and running the adapter are located at the following Web site: IBM WebSphere Adapters and IBM WebSphere Business Integration Adapters: software requirements. From the IBM WebSphere Adapters list, select the link for the Adapter for JDBC, Version 6.0.2.

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. The WebSphere Adapters software is 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.

Installation

You can install WebSphere Adapters either through a graphical user interface or silently through a script. The silent installation method is recommended for users with accessibility needs.

Administration

The administrative console of either WebSphere Process Server or WebSphere Enterprise Service Bus is the primary interface for deployment and administration of the enterprise applications. These consoles are 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 using 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.

Enterprise service discovery wizard

The enterprise service discovery wizard is the primary component used to create enterprise applications with the adapters. This wizard is implemented as an Eclipse plug-in that is available through WebSphere Integration Developer and is fully accessible.

Keyboard navigation

This product uses standard Microsoft Windows® navigation keys.

IBM and accessibility

See the *IBM Accessibility Center* for more information about the commitment that IBM has to accessibility.

Internet Protocol Version 6.0

IBM WebSphere Process Server relies on WebSphere Application Server for Internet Protocol Version 6.0 compatibility.

IBM WebSphere Application Server Version 6.0 and its JavaMail component support dual stack Internet Protocol Version 6.0 (IPv6).

For more information about this compatibility in WebSphere Application Server, see IPv6 support in the WebSphere Application Server information center.

For more information about IPv6, see www.ipv6.org.

Technical overview of the Adapter for JDBC

The Adapter for JDBC supports integration of databases that are accessible through the JDBC application programming interface (API) by providing inbound and outbound processing under the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA).

Under *outbound* operations, a business object is passed to the adapter as a request that is processed according to the operation specified in the business object, either create, retrieve, update, delete, retrieveall, or execute. Requests are received from different EIS applications that need to have the updates applied to the database being managed by the adapter. Processing of these requests results in the creation, retrieval, update or deletion of rows in the corresponding database tables.

Inbound operations can either use standard event processing with an event store or can use custom event processing.

During *standard event processing*, as data is changed in the application tables in the database, appropriate events are inserted into an event table called an event store, along with relevant information, such as key values. To capture the changed data, you can place triggers on the respective tables, or use other methods, such as Oracle Change Data Capture, which is provided for Oracle databases. The Adapter for JDBC polls the event store and retrieves a batch of events. The events can be filtered based on business object type and/or timestamp. These events are processed, and each event is used to construct a JDBC business graph. The business graph is then dispatched to the endpoints that have a subscription for the specific business object.

During *custom event processing*, you can enter a query as either a standard SQL statement, a stored procedure or a stored function. Any of these actions returns a resultset that has the data for the following database columns in order:

- event_id
- object_key
- object_name
- object_function.

The adapter constructs a JDBC business graph of each event and delivers it to the endpoints that have a subscription for the specific business object. The adapter also supports custom update and delete queries for custom event processing. The custom update and delete queries are run after each event is processed. The queries take the event_id as an input parameter. The update query ensures that the same record does not get picked up for processing during subsequent poll cycles.

The figure titled "Processing within the JDBC adapter" shows inbound and outbound operations.

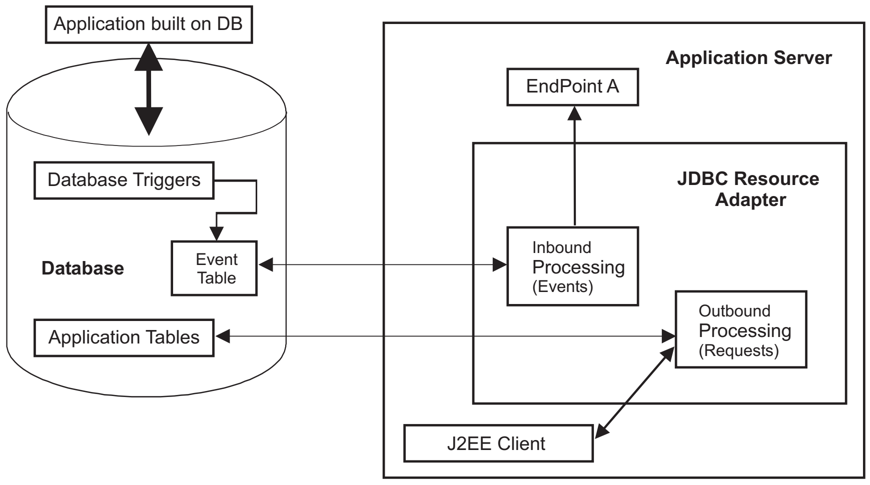


Figure 2. Processing within the Adapter for JDBC

Outbound processing

When an appplication component needs to invoke an operation on the enterprise information system (EIS), for example to retrieve data, the Adapter for JDBC acts as the connector between the application component and the EIS. The adapter provides certain outbound operations. The standard outbound operations can process either after-image or delta style business objects. The adapter also supports both local and XA transactions for outbound operations.

Outbound operations

The adapter supports two business object styles that relate to the amount and purpose of information conveyed by the business object: after-image and delta. An *after-image* is the state of a business object after all changes have been made to it. A *delta* is a business object used in an update operation that contains only key values and the data to be changed.

The adapter provides after-image support for the following operations:

- Create
- Update
- Delete

The adapter provides delta support for the following operation:

- ApplyChanges

Outbound operations without after-image or delta support::

- Retrieve
- RetrieveAll
- Execute (available only with stored procedures)

Verbs are definable only for after-image business objects. A *verb* reflects the state of the business object, whereas an *operation* reflects the operation to be performed by the adapter. The top-level verbs supported for after-image are:

- Create

- Update
- Delete
- UpdateWithDelete

For details on how the adapter processes business objects for each of the supported outbound operations, see the “Outbound operations” section.

Transaction management

The Adapter for JDBC supports both local and XA transactions. In the adapter, a transaction is an isolated interaction with the back-end database, or enterprise information system (EIS). A transaction can consist of multiple operations on the database that are performed as an atomic unit. These operations are not affected by simultaneously occurring operations from other client applications of the database.

The Adapter for JDBC supports transactions only if the back-end database supports transactions. The types of transactions that are supported are local and XA transactions:

- In a *local transaction*, a given client application defines the start and end of the transaction with the database. It uses a one-phase-commit protocol.
- In an *XA transaction*, the transaction spans multiple heterogeneous databases. It uses a global or two-phase-commit protocol.

XA transactions for IBM DB2 and Oracle databases

The adapter supports XA transactions for IBM DB2 and Oracle databases only.

Use the properties XADataSourceName and XADatabaseName with XA transactions. See the “Managed (J2C) connection factory properties” in the “Reference” section for details about these properties.

Sample values for two XA transaction properties are provided below:

IBM DB2

XADataSourceName for Type 2 Driver (db2java.zip):
`COM.ibm.db2.jdbc.DB2XADataSource`

XADataSourceName for Type 4 Driver (db2jcc.jar):
`com.ibm.db2.jcc.DB2XADataSource`

XADatabaseName : `<dbname>`
where *dbname* is the name of the database.

Oracle

XADataSourceName: `oracle.jdbc.xa.client.OracleXADataSource`

Note: If you are using a DB2 database, you must configure the XADatabaseName property. If you are using an Oracle database, no value is required for the XADatabaseName property.

Transaction support with DataSourceJNDIName

If the DataSourceJNDIName property is defined as a local or XA transaction data source, then the adapter uses this property for obtaining the inbound or outbound connection to the database. The DataSourceJNDIName represents the DataSource created within WebSphere Process Server or WebSphere Enterprise Service Bus. This name could represent an XA or Connection Pool data source.

If UserName and Password properties are provided, these are used along with DataSourceJNDIName to obtain the connection. If UserName and Password are not provided, the adapter uses the username and password that were set when DataSource was created within the server. All other properties, such as JDBCDriverClass, DatabaseURL and the XA properties, XA DataSourceName and XADatabaseName, are ignored.

Important: XA transaction support is not restricted to IBM DB2 and Oracle databases if you use XA data sources through the DataSourceJNDIName property. The database restriction applies to XA transaction support using non-data source connection to a database.

Inbound processing

The Adapter for JDBC supports inbound event management with asynchronous event delivery. Events are processed by using these two modes: a standard event table and custom queries.

Asynchronous event delivery is accomplished through the use of:
- A standard event table, called an event store
- Custom event processing

Standard event table

For any changes in the user tables, the application populates the event table, called an event store, in the enterprise information system (EIS). Updates are made by placing triggers on the user tables that record events in the event table corresponding to the updates to the user table.

If you set the Activation specification property AssuredOnceDelivery to true, an xid (transaction id) value is set for each event in the event store. After an event is obtained for processing, the xid value for that event is updated in the event table. The event is then delivered to its corresponding endpoint, and subsequently deleted from the event table. If the database connection is lost or the application is stopped before the event can be delivered to the endpoint, the event may not be processed completely. In this case, the xid column ensures that the event is reprocessed and sent to the endpoint. Once the database connection is re-established or the adapter starts again, the adapter checks for events in the event table that have a value in the xid column. The adapter processes these events first, and then polls the other events during the poll cycles. The event table is described below.

The adapter can filter the events to be processed by business object type. The filter is set by use of the Activation specification property EventFilterType. This property has a comma-delimited list of business object types. Only the types specified in the property are processed. If no value is specified for the property, no filter is applied, and all the events are processed. If the Activation specification property FilterFutureEvents is set to true, the adapter filters events based on timestamp. The adapter compares the system time in each poll cycle to the timestamp on each event. If an event is set to occur in the future, it will not be processed until that time.

Custom event processing

Custom queries can be entered by using:
- Standard SQL
- A stored procedure
- A stored function

All of these entries take an input parameter for PollQuantity, an Activation specification property that the adapter provides at run time. These queries return a resultset, with the poll quantitiy number of records, that has the following columns in order: event_id, object_key, object_name, and object_function. The columns represent the following information:
- **event_id**: A unique id for each event; this could be same as the object_key value.
- **object_key**: The key value of the record in the table that will be retrieved for event processing.
- **object_name**: The name of the business graph generated using enterprise service discovery. The business object within the business graph can be a hierarchical business object. Each business object refers to a table or view.
- **object_function**: The Create, Update, or Delete operation to be set on the event.

The three types of custom queries are described as follows.

Standard SQL

The user can enter a SQL select statement that takes in one input parameter for poll quantity. The query can have other input parameters as well, but the values for all those parameters need to be set in the SQL query itself. The SQL query returns a resultset (with poll quantity number of records) that should have the following columns in order: event_id, object_key, object_name, and object_function. The Adapter for JDBC generates the event object and processes the events.

Stored procedure

The custom query can be a stored procedure that returns a result set. The adapter passes the poll quantity value when the stored procedure is called. So, the user-entered procedure should accept an input parameter for the poll quantity. Also, the stored procedure has an output parameter of type resultset. The resultset returned by the stored procedure should have the following columns in order: event_id, object_key, object_name, and object_function. Following is the syntax to use when specifying a stored procedure:

```
call <sp_name> (?, ?)
```

where sp_name is the name of the stored procedure that should be run.

Note: The first parameter represents the poll quantity, and the second parameter represents the resultset.

The stored procedure can take in other input parameters as well, but you need to provide values for those input parameters in the call statement itself, for example:
```
call <sp_name> (25, ?, ?)
```

Stored function

The custom query can also be a stored function that returns a result set. The adapter passes the poll quantity value when the function is called. So, the user-entered function should accept an input parameter for the poll quantity. Also, the return value for the function should be a resultset. The resultset returned by the function should have the following columns in order: event_id, object_key, object_name, and object_function. Following is the syntax to use when specifying a stored function:

? = <*sp_name*> (?)

where sp_name is the name of the stored function that should be run.

Note: The first parameter represents the resultset, and the second parameter represents the poll quantity.

The stored function can take in other input parameters but you need to provide values for those input parameters in the call statement itself, for example: ? = call <*sp_name*> (25, ?)

The adapter also provides support for custom update and delete queries. Users would generally use an *update query* to ensure that the same record does not get picked up for processing during subsequent poll cycles. The *delete query* is used in case records need to be deleted after each event is processed. Both the update and delete queries are optional.

These queries, if specified in the Activation specification, are run after each event is processed. The user can enter these queries in two ways: either as a standard SQL statement or as a stored procedure. The stored procedure syntax is the same as the one specified for the custom query Activation specification property. Both the standard SQL and the stored procedure take an input parameter for the event id. The adapter provides the value of event id at runtime. The queries can also have additional input parameters, but those need to be provided in the query syntax itself as described for the custom event query.

Custom queries support assured once delivery if you have created the standard event store for storing xid values. The adapter stores the events returned by the custom event query in the event store, and it updates the events with xid values. The adapter processes the events in the same way as for standard event table processing. Since the standard event store is used for custom event processing with the AssuredOnceDelivery property set to true, a custom query cannot query on the standard event table. In addition, in this scenario the event table should not have an automatic generation of event id values, because the adapter populates the event id value it retrieves from the custom query in the event table. With custom queries, no support is provided for event filtering.

Event table

The event table is described in the following table.

Table 1. Event table

Name	Type	Description
xid	String	Unique transaction id (xid) value for assured once delivery

Table 1. Event table (continued)

Name	Type	Description
event_id	Number	Unique Event id that is a Primary key for the table
object_key	String	Delimited string that contains keys for the business object (not null)
object_name	String	Name of the business object (not null)
object_function	String	Operation corresponding to the event (Delete, Create, Update, and so on) (not null)
event_priority	Number	Any positive integer value (not null)
event_time	Timestamp	Date and time when event was generated
event_status	Number	The values are as follows: • 0 Ready for poll • 3 In progress • -1 Event processing failed (not null)
event_comment	String	Description

Business objects

Business data that is exchanged in an integrated application in WebSphere Process Server or WebSphere Enterprise Service Bus is represented by constructs called business objects. The business object is a fundamental data structure for representing business data.

Business objects are based upon the open-standard Service Data Objects (SDO) programming model, which supports common data access. The WebSphere adapter business object provides extensions and functionality in addition to the SDO model. In Service Component Architecture (SCA), business objects represent the data that flows between service components in integrated applications.

Business object naming conventions

Business object names should reflect the structure they represent, such as Customer or Address. Names are normally derived during the metadata import process of enterprise metadata discovery, based on the name given by the enterprise information system (EIS).

Enterprise metadata discovery replaces any special character in the business object name with `U` followed by its Unicode number, except for names that have the underscore (_) character. For example, the business object name would be Order_Item for the Order_Item table in the database application and ShippingU45Address for the Shipping-Address table.

The parent business object graph should be named for the contained business object, followed by BG; for example, CustomerBG for a Customer business object.

Business object names have no semantic value to the adapter or the database; that is, they derive no information nor meaning from the business object name. If one name is replaced by another, the adapter behavior remains the same.

Business objects carry database-specific metadata. A name can use a string like JDBC or %*AppName*% as a prefix to help distinguish between the two types of business objects: application-specific and generic. The remainder of the name can describe the table or stored procedure that the business object represents. For example, if the business object definition is generated for the Employee Table in a database application, such as Human Resources (HR), the respective business object name will be HREmployee.

An example of business object names not derived through the metadata import process are names for query business objects that you specify by using the enterprise service discovery wizard. Avoid specifying duplicate names for different query business objects, otherwise the names will be overwritten.

You can rename business objects by using the refactoring functionality in WebSphere Integration Developer. For more details, refer to the WebSphere Integration Developer documentation.

Business object structure

The Adapter for JDBC supports table and view business objects. For applications that use databases, each business object corresponds to a database table or view, and each simple attribute within the object corresponds to a column in that table or view. In addition, the adapter supports stored procedure and query business objects.

For table or view business objects

A *simple attribute* is an attribute that represents a single value, such as a String, Integer, or Date. Thus, attributes within the same business object cannot be stored in different database tables; however, the following situations are possible:

- The database table might have more columns than the corresponding business object has simple attributes; that is, some columns in the database are not represented in the business object. Only those columns needed for processing of the business object should be included in your design.
- The business object might have more simple attributes than the corresponding database table has columns; that is, some attributes in the business object are not represented in the database. The attributes that do not have a representation in the database either have no application-specific information, are set with default values, or are specified by stored procedures.
- The business object can represent a view that spans multiple database tables. The adapter can use such a business object when processing requests triggered in the application, such as Create, Update, and Delete operations. When processing business object requests, however, the adapter can use such a business object only for Retrieve and RetrieveAll requests.

For stored procedure business objects

In this case, business objects are generated for stored procedures. All of the stored procedure input and output parameters have corresponding attributes in the business object. If any of the input or output parameters is of a complex type, such as an array or struct, then the corresponding business object attribute is a child business object type with the child business object containing the attributes of the array or struct. If the stored procedure returns a result set, a child business object is created that contains the attributes of the returned result set.

Two examples are given below to show the structure of stored procedure business objects. The business objects, ScottStrtValues and ScottStrtvaluesStrt, are generated from a stored procedure that has one input type and two output types. One of the output parameters is of the struct data type. Enterprise metadata discovery generates a business object, ScottStrtValuesStrt, for the struct type and adds it as a child object to the parent business object, ScottStrtValues. For the attribute of type struct in the parent business object, the ChildBOType application-specific information is set to Struct to indicate it is of type struct. The ChildBOTypeName application-specific information is set to the value of the user-defined struct type in the database.

Example of ScottStrtValues business object

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvalues" xmlns:scottstrtvalues=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvalues" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
metadata" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt">
<import namespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt"
schemaLocation="ScottStrtvaluesStrt.xsd"/>
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvalues">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
<jdbcasi:MaxNumOfRetRS>0</jdbcasi:MaxNumOfRetRS>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>IP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="strt" type="scottstrtvaluesstrt:ScottStrtvaluesStrt"
minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
```

```
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
<jdbcasi:ChildBOType>STRUCT</jdbcasi:ChildBOType>
<jdbcasi:ChildBOTypeName>STRUCT1</jdbcasi:ChildBOTypeName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

Example of ScottStrtValuesStrt business object

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvaluesStrt">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="name" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="title" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="dept_num" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
```

```
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

For query business objects

An example select statement is provided to show the structure of a query business object. For the following select statement:

```
SELECT C.custid, C.custname, A.phone FROM customer C, address A WHERE
(C.custid = A.custid) AND (C.custname LIKE ?)
```

The query business object has the following format:

Business object level application-specific information as follows:

```
SelectStatement = SELECT C.custid, C.custname, A.phone FROM customer C,
address A WHERE (C.custid = A.custid) AND (C.custname LIKE ?)
```

The attributes as follows:
- custid ColumnName=CUSTID
- custname ColumnName=CUSTNAME
- phone ColumnName=PHONE
- parameter1 (Corresponding to the one ?, should have as many parameters as there are ?s)
- jdbcwhereclause

The operation supported: RetrieveAll

The following example shows a query business object definition file.

Example query business object

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/querybo1" xmlns:querybo1=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/querybo1" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="QueryBO1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SelectStatement>select * from customer where
pkey=?</jdbcasi:SelectStatement>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="string" minOccurs="0" maxOccurs="1">
<annotation>
```

```
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>PKEY</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>FNAME</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="lname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>LNAME</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="ccode" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>CCODE</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="parameter1" type="string" minOccurs="1" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>Parameter1</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="jdbcwhereclause" type="string" minOccurs="0" maxOccurs="1"
default="where pkey=?">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

For all business objects

Business objects can be either flat or hierarchical. All of the attributes of a flat business object are simple and represent one row in the database table. The term *hierarchical* business object refers to a complete business object, including all the child business objects that it contains at any level. The term *individual* business object refers to one business object, independent of child business objects that it might contain or that contain it. The individual business object can represent a view that spans multiple database tables. The term *top-level* business object refers to the individual business object at the top of the hierarchy that does not itself have a parent business object.

A hierarchical business object has attributes that represent a child business object, an array of child business objects, or a combination of the two. In turn, each child business object can contain a child business object or an array of business objects, and so on.

A *single-cardinality relationship* occurs when an attribute in a parent business object represents one child business object. In this case, the attribute is of the same type as the child business object.

A *multiple-cardinality relationship* occurs when an attribute in the parent business object represents an array of child business objects. In this case, the attribute is of the same type as the child business objects.

The adapter supports the following relationships among business objects:
- Single-cardinality relationships
- Single-cardinality relationships and data without ownership
- Multiple-cardinality relationships

In each type of cardinality, the relationship between the parent and child business objects is described by the application-specific information of the key attribute in the business object storing the relationship. For more information on this application-specific information, see “ForeignKey” in the “Attribute application-specific information” table.

Single-cardinality relationships in business objects:

In a single-cardinality relationship, an attribute in a parent business object represents one child business object. In this case, the attribute is of the same type as the child business object. The adapter supports single-cardinality relationships, and single-cardinality relationships and data without ownership.

Single-cardinality relationships

Typically, a business object that contains a single-cardinality child business object has at least two attributes that represent the relationship. The type of one attribute is the same as the child’s type. The other attribute is a simple attribute that contains the child’s primary key as a foreign key in the parent. The parent has as many foreign key attributes as the child has primary key attributes.

Because the foreign keys that establish the relationship are stored in the parent, each parent can contain only one child business object of a given type.

The figure titled “Typical single-cardinality relationship” illustrates a typical single-cardinality relationship. In the example, FKey in the ParentBOName box is the simple attribute that contains the child’s primary key, and Child(1), also in the ParentBOName box, is the attribute that represents the child business object.

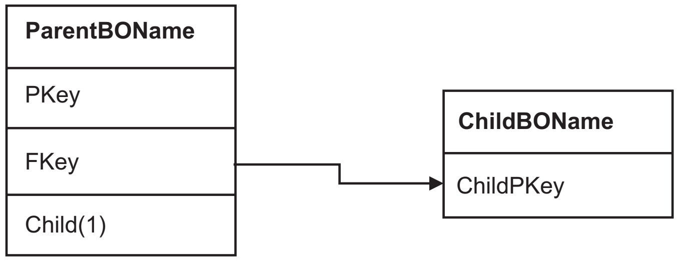


Figure 3. Typical single-cardinality relationship

A parent business object can have a single-cardinality OWNERSHIP child and a single-cardinality NOOWNERSHIP child. Lookup tables are used for NOOWNERSHIP relationships.

Single-cardinality relationships and data without ownership

Typically, each parent business object owns the data within the child business object that it contains. For example, if each Customer business object contains one Address business object, when a new customer is created, a new row is inserted into both the customer and address tables. The new address is unique to the new customer. Likewise, when deleting a customer from the customer table, the customer’s address is also deleted from the address table.

However, situations can occur in which multiple hierarchical business objects contain the same data, which none of them owns. For example, assume that an Address business object has a *StateProvince[1]* attribute that represents the *StateProvince lookup table with single cardinality*. Because the lookup table is rarely updated and is maintained independently of the address data, creation or modification of address data does not affect the data in the lookup table. The adapter either finds an existing state or province name or fails. It does not add or change values in the lookup table.

When multiple business objects contain the same single-cardinality child business object, the foreign key attribute in each parent business object must specify the relationship as `NOOWNERSHIP`. When an application server sends the adapter a hierarchical business object with a Create, Delete, or Update request, the adapter ignores single-cardinality children contained without ownership. The adapter performs only retrieve operations on these business objects. If the adapter fails to retrieve such a single-cardinality business object, it returns an error and stops processing.

Denormalized data and data without ownership

In addition to facilitating the use of static lookup tables, containment without ownership provides another capability: synchronizing normalized and denormalized data.

Synchronization of normalized to denormalized data: Specifying a relationship as `NOOWNERSHIP` enables you to create or change data when you synchronize from a normalized application to a denormalized one. For example, assume that a normalized source application stores data in two tables, A and B. Assume further that the denormalized destination application stores all the data in one table such that each entity A redundantly stores B data.

In this example, to synchronize a change in table B data from the source application to the destination application, you must trigger a table A event whenever table B data changes. In addition, because table B data is stored redundantly in table A, you must send a business object for each row in table A that contains the changed data from table B.

Note: When making updates to denormalized tables, ensure that each record has a unique key so that multiple rows are not modified as a result of one update. If such a key does not exist, the Adapter for JDBC provides an error stating that multiple records have been updated.

Synchronization of denormalized to normalized data: When synchronizing data from a denormalized source application to a normalized destination application, the adapter does not create, delete, or update data contained without ownership in the normalized application.

When synchronizing data to a normalized application, the adapter ignores all single-cardinality children contained without ownership. To create, remove, or modify such child data, you must process the data manually.

Multiple-cardinality relationships in business objects:

In a multiple-cardinality relationship, an attribute in the parent business object represents an array of child business objects. The attribute is of the same type as the child business object. The foreign key that describes the relationship is stored in the child, except when an application stores a single-child entity. Then the parent-child relationship is stored in the parent.

Typically, a business object that contains an array of child business objects has only one attribute that represents the relationship, and this attribute is normally the primary key. The type of the attribute is an array of the same type as the child business objects. For a parent to contain more than one child, the foreign keys that establish the relationship are stored in the child.

Therefore, each child has at least one simple attribute that contains the parent's primary key as a foreign key. The child has as many foreign key attributes as the parent has primary key attributes.

Because the foreign keys that establish the relationship are stored in the child, each parent can have zero or more children.

The figure titled "Multiple-cardinality business object relationship" illustrates a multiple-cardinality relationship. In the example, ParentId in the three ChildBOName boxes is the simple attribute that contains the parent's primary key, and Child1 in the ParentBOName box is that attribute that represents the array of child business objects.

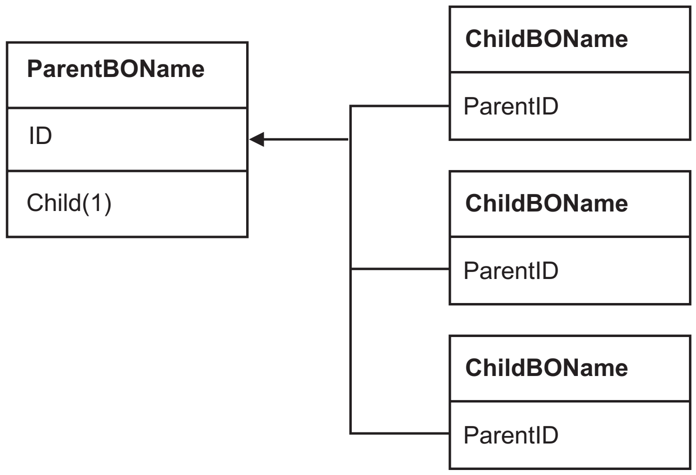


Figure 4. Multiple cardinality business object relationship

A multiple cardinality relationship could be an N=1 relationship. Some applications store one child entity so that the parent-child relationship is stored in the child rather than in the parent. In other words, the child contains a foreign key whose value is identical to the value stored in the parent's primary key.

Applications use this type of relationship when child data does not exist independently of its parent and can be accessed only through its parent. Such child data requires that the parent and its primary key value exist before the child and its foreign key value can be created. The figure titled "Multiple cardinality relationship with N=1" shows this type of relationship.

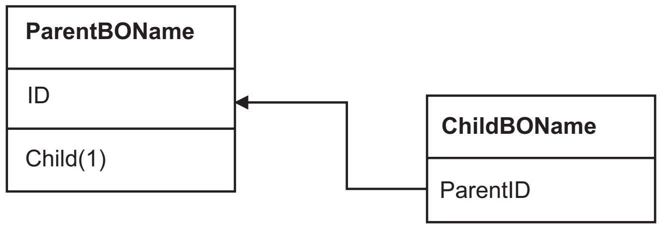


Figure 5. Multiple cardinality relationship with N=1

Business object attribute properties

Business objects contain attributes that are used to define the content of a business object. Each attribute has a name, type, cardinality and several other properties. A business object is simply a container for the data specified in its attributes.

The following table titled "Attribute properties" lists the properties and their interpretation and settings.

Table 2. Attribute properties

Properties	Interpretation and settings
Cardinality	Each business object attribute that represents a child or an array of child business objects has the value of single (1) or multiple (n) cardinality, respectively.

Table 2. Attribute properties (continued)

Properties	**Interpretation and settings**
Foreign Key	When arrays of child business objects whose cardinality is *n* are retrieved, foreign keys are used in the WHERE clause of SELECT statements. **Note:** The adapter does not support specifying an attribute that represents a child business object as a foreign key.The RetrieveAll operation overrides the use of keys and foreign keys.
Primary Key	At least one simple attribute in each business object must be specified as the primary key. **Note:** The adapter does not support specifying an attribute that represents a child business object or an array of child business objects as a primary key attribute. If the primary key property is set to `true` for a simple attribute, the adapter adds that attribute to the WHERE clause of the SELECT statement and UPDATE SQL statements that it generates while processing the business object. The RetrieveAll operation overrides the use of primary and foreign keys.
Name	This property represents the unique name of the attribute, if it is a simple attribute, or the name of the business object, if it is a child business object.
Required	Specifies whether an attribute must contain a value. If this property is set to `true` for a container whose cardinality is single (1), then the adapter requires that the parent business object contain a child business object for this attribute. Business objects that are passed to the adapter for Create, Update, and Delete operations must also contain a child business object. Cardinality is single (1) for simple attributes and multiple (n) for container attributes. The adapter causes a Create operation to fail if a business object does not have a valid value or a default value for a required attribute. It also fails if no data is available upon retrieval from the database for this object.
Type	The type of the attribute (such as Integer, String, Date, Timestamp, Boolean, Double, or Float) if it is a simple attribute, or the type of business object if it is a child business object. When the adapter encounters an attribute of a type that it does not support, the adapter wraps the value in quotation marks and handles the value as character data.

Outbound operations

Application components need to invoke operations such as data retrieval on the enterprise information system. The Adapter for JDBC provides certain outbound operations. Details are provided on how the adapter processes business objects for each of the supported operations.

Create operation

When given a hierarchical business object, the create operation recursively traverses the business object, creating rows corresponding to each table.

Here are the details:

1. The create operation recursively inserts each single-cardinality child business object contained with ownership into the database. In other words, the adapter creates the child and all child business objects that the child and its children contain.

If the business object definition specifies that an attribute represents a child business object with single cardinality and that attribute is empty, the adapter ignores the attribute. However, if the business object definition requires that the attribute represent a child, and it does not, the adapter returns an error and stops processing.

2. The create operation retrieves and checks the existence of each single-cardinality child business object contained without ownership. If the retrieve operation is unsuccessful, indicating that the child does not exist in the database, the adapter returns an error and stops processing. If the retrieve operation is successful, the adapter recursively updates the child business object.

   Note: For this approach to work correctly when the child business object exists in the application database, primary key attributes in child business objects must be cross-referenced correctly on create operations. If the child business object does not exist in the application database, the primary key attributes must not be set.
3. It inserts the top-level business object in the database as follows:
   a. It sets each of its foreign key values to the primary key values of the corresponding child business object represented with single cardinality. Because values in child business objects can be set by database sequences or counters or by the database itself during the creation of the child, this step ensures that the foreign key values in the parent are correct before the adapter inserts the parent in the database.
   b. It generates a new, unique ID value for each attribute that is set automatically by the database. The name of the database sequence or counter is stored in the attribute's application-specific information. If an attribute has an associated database sequence or counter, the value generated by the adapter overwrites any value passed in by the application server. For more information on specifying a database sequence or counter, see UID=AUTO in "Application-specific information for simple attributes."
   c. It inserts the top-level business object into the database.
4. It processes each of its multiple-cardinality child business objects as follows:
   a. It sets the foreign key values in each child to reference the value in the corresponding primary key attributes in the parent. Because the parent's primary key values might have been generated during the creation of the parent, this ensures that the foreign-key values in each child are correct before the adapter inserts the child into the database.
   b. It inserts each of its multiple-cardinality child business objects into the database.

Retrieve operation

This topic describes the steps the adapter takes to retrieve a hierarchical business object.

The adapter performs the retrieve operation as follows:

1. It removes all child business objects from the top-level business object received. In other words, it makes a copy of the top-level business object without any children.
2. It retrieves the top-level business object from the database.
   - If the retrieval returns one row, the adapter continues processing.

- If the retrieval returns no rows, indicating that the top-level business object does not exist in the database, the adapter returns the error `RecordNotFoundException`.
- If the retrieval returns more than one row, the adapter returns an error.

3. It recursively retrieves all multiple-cardinality child business objects.

   Note: The adapter does not enforce uniqueness when populating an array of business objects. It is the database's responsibility to ensure uniqueness. If the database returns duplicate child business objects, the adapter returns duplicate children.
4. It recursively retrieves each of the single-cardinality children, regardless whether the child business object is contained with or without ownership.

   Note: All single-cardinality child business objects are processed based on their occurrence in the business object and before the parent business object is processed. Child object ownership and non-ownership do not determine the processing sequence, but they do determine the type of processing.

RetrieveAll operation

The adapter uses the RetrieveAll operation to retrieve an array of business objects from the database. The process differs depending on whether the RetrieveAll operation is for database table business objects or for user-specified SQL business objects.

For database table business objects

All of the key and non-key attributes populated in the incoming business object determine the selection criteria. The adapter may retrieve multiple rows for the top-level business object from the database, depending on the attributes selected. If no attributes are populated in the incoming business object, all the rows are retrieved from the respective table in the database.

The adapter performs the following steps to retrieve an array of business objects:

1. For each of the rows retrieved from the database, the adapter constructs a top-level business graph and creates a container of business graphs using all of the retrieved rows. The name of the container business object is *BOName + ContainerBG*.
2. The adapter retrieves each of the business graphs in the container using the Retrieve operation.

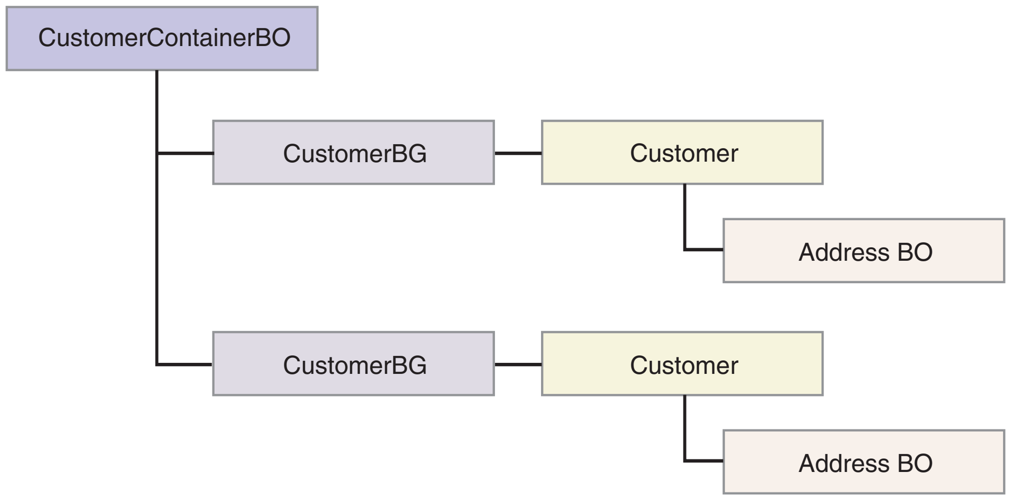


Figure 6. Structure of the business object returned in a RetrieveAll operation

The following errors can result from a RetrieveAll operation:

- If any populated business object in the input object does not exist in the EIS, the adapter returns the error `RecordNotFoundException`.
- If the number of hits in the EIS exceeds the value of ResultSetLimit defined in the interaction specification, the adapter returns the error MatchesExceededLimitException. The MatchCount property contains the actual number of hits that the adapter had in the EIS, so that you can either increase the limit, or refine the search appropriately.

  Note: If ResultSetLimit is set to a very large number, problems may occur related to a lack of sufficient memory, depending upon the size and number of business objects returned.
- If any unrecoverable errors are reported by the EIS, the adapter returns the error EISSystemException.

For user-specified SQL business objects

Business objects that are created for user-specified SQL statements also support the RetrieveAll operation. When the enterprise service discovery (ESD) wizard generates the query business object for the user-specified SQL statement, the adapter runs the SQL SELECT statement, and returns data to the database in the structure shown in the following figure.

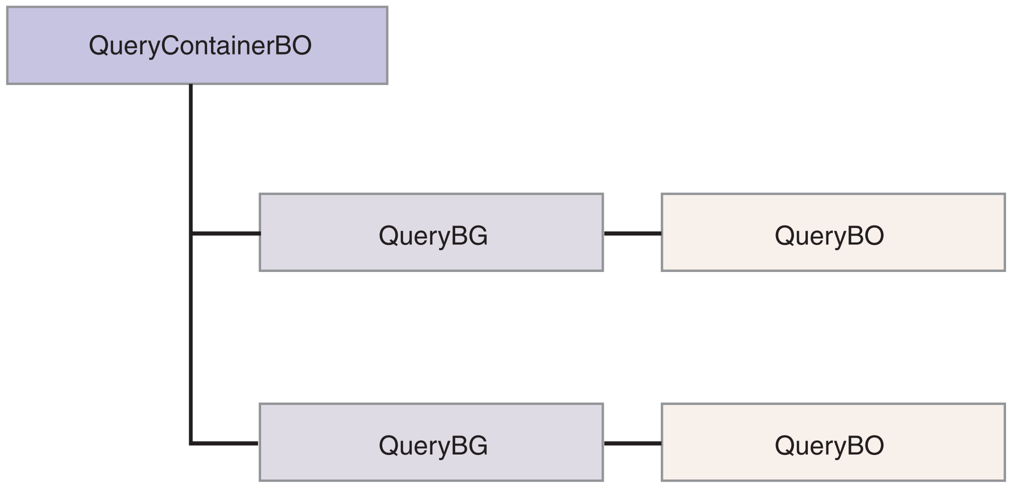


Figure 7. User-specified SQL business objects

To process the query business object generated by ESD for the user-specified SELECT statement, the adapter:

1. Obtains the SELECT SQL statement from the query business object.
2. Checks whether a dynamic where clause is specified in the query business object.
   - If there is a dynamic where clause, the adapter replaces the default where clause in the SELECT statement with the dynamic one.
   - If there is no dynamic where clause, the adapter replaces parameters in the SELECT statement with the corresponding values specified in the query business object.
3. Runs the SELECT statement
4. Obtains the result set that is returned and populates the query business object values with the data returned from the database, creating a container business object with the structure shown in the previous figure.
5. Does a deep retrieve of each top-level query business object in the container, depending on whether any child business objects are defined for the query business object.

   Note: Query business objects can be only top-level business objects. A query business object cannot have child query business objects.

Update operation

The update operation is done by comparing the incoming business object with a business object that is retrieved from the database using the primary keys specified in the top-level, incoming business object.

The adapter performs the following steps when updating a hierarchical business object:

1. It uses the primary key values of the source business object to retrieve the corresponding entity from the database. The retrieved business object is an accurate representation of the current state of the data in the database.

   If the retrieval fails, indicating that the top-level business object does not exist in the database, the adapter returns a `RecordNotFoundException` error, and the update fails.

If the retrieval succeeds, the adapter compares the retrieved business object to the source business object to determine which child business objects require changes in the database. The adapter does not, however, compare values in the source business object's simple attributes to those in the retrieved business object. The adapter updates the values of all non-key simple attributes.

If all of the simple attributes in the top-level business object represent keys, the adapter cannot generate an update query for the top-level business object. In this case, the adapter logs a warning and continues to step 2.

2. It recursively updates all single-cardinality children of the top-level business object.

   If the business object definition requires that an attribute represent a child business object, the child must exist in both the source business object and the retrieved business object. If it does not, the update operation fails, and the adapter returns an error.

   The adapter handles single-cardinality children contained with ownership in one of the following ways:

   - If the child is present in both the source and the retrieved business objects, instead of updating the existing child in the database, the adapter deletes the existing child and creates the new child.
   - If the child is present in the source business object but not in the retrieved business object, the adapter recursively creates it in the database.
   - If the child is present in the retrieved business object but not in the source business object, the adapter recursively deletes it from the database.

   For single-cardinality children contained without ownership, the adapter attempts to retrieve every child from the database that is present in the source business object. If it successfully retrieves the child, the adapter populates the child business object but does not update it, because single-cardinality children contained without ownership are never modified by the adapter.
3. It updates all simple attributes of the retrieved business object, except those whose corresponding attribute in the source business object is not specified.

   Because the business object being updated must be unique, the adapter verifies that only one row is processed as a result. It returns an error if more than one row is returned.
4. It processes each multiple-cardinality child of the retrieved business object in one of the following ways:
   - If the child exists in both the source and the retrieved business objects' arrays, the adapter recursively updates it in the database.
   - If the child exists in the source array but not in the retrieved business object's array, the adapter recursively creates it in the database.
   - If the child exists in the retrieved business object's array but not in the source array, the adapter recursively deletes it from the database unless the application-specific information for the attribute that represents the child in the parent has KeepRelationship set to `true`. In this case, the adapter does not delete the child from the database.

DeltaUpdate operation

If the operation in the InteractionSpec is ApplyChanges, and the verb does not exist in the business graph, the adapter performs the DeltaUpdate operation. The adapter inspects the ChangeSummary to identify the operation for each business object in the input hierarchy and performs the operation that was identified.

DeltaUpdate operations are different from Update operations as follows:

- In a DeltaUpdate operation, no retrieve operation occurs before updating.
- No comparisons are made between the incoming business object and the business object in the database.
- All children are processed based on the verb set in each child object. If a child does not have a verb set in it, the adapter returns an error.

The adapter performs the following steps when updating a hierarchical business object with DeltaUpdate. It processes only object changes from the ChangeSummary:

1. It recursively processes all single-cardinality children of the parent object. If a child is marked required in the business object specification, it must be present in the inbound object. If it is not, the DeltaUpdate operation fails, and the adapter returns an error.
2. It sets all foreign key values in the parent that reference attributes in single-cardinality children to their corresponding child values. This is necessary because single-cardinality children might have been added to the database during the previous steps, resulting in the generation of new sequence values.
3. It updates the current object being processed using an SQL Update statement or a stored procedure. All simple attributes of the individual business object are updated. The adapter does not use property level changes to determine which attributes need to be added to the update statement; they are all updated. Because the object being updated should be unique, the adapter checks to ensure that only one row is processed as a result. An error is returned if more than one row is processed.
4. It sets all foreign key values in all cardinality N children of the current object that reference parent attributes to the corresponding parent values. Usually these values are already cross-referenced during data mapping; however, this might not be the case for new children in cardinality N containers. This step ensures that the foreign-key values in all cardinality N children are correct before those children are updated.
5. It updates all cardinality N containers of the current object.

   When the child objects are processed, each child's verb is taken and the appropriate operation is performed. The allowed operations on a child in DeltaUpdate are create, delete, and update:

   - If a Create verb is found in the child, the child is created in the database if it is an ownership child. Non-ownership children are retrieved to validate their existence in the database.
   - If a Delete verb is found in the child, that child is deleted.
   - If an Update verb is found in the child, the child gets updated in the database.

Delete operation

The delete operation is performed by pruning the incoming business object and then retrieving the complete business object from the database. The delete operation is then applied recursively on each business object in the hierarchy.

The delete operation supports physical and logical deletes, depending on the StatusColumnName value in the object's application-specific information. If the StatusColumnName value is defined, the adapter performs a logical delete operation. If the StatusColumnName value is not defined, the adapter performs a physical delete operation.

For physical deletes the adapter takes the following actions:

- It recursively deletes all multiple-cardinality child business objects.
- It deletes the top-level business object.
- It recursively deletes all single-cardinality child business objects contained with ownership.

For logical deletes the adapter takes the following actions:

- It issues an Update that sets the business object's status attribute to the value specified by the business object's application-specific information. The adapter ensures that only one database row is updated as a result, and it returns an error if this is not the case.
- It recursively logically deletes all single-cardinality children contained with ownership and all multiple-cardinality children. The adapter does not delete single-cardinality children contained without ownership.

ApplyChanges operation

The ApplyChanges operation covers a variety of things. It enables any business object requiring a create, update, or delete operation to be processed accordingly by the adapter.

If the top-level verb exists in the business object, then the business object is processed as an after-image. If no top-level verb exists in the business object, then the ChangeSummary is processed.

Execute operation

The Execute operation is used to run stored procedures. The enterprise service discovery wizard generates the required stored procedure business object that corresponds to the stored procedure definition in the database. The adapter uses the Execute operation to process the stored procedure business object.

The following information provides a simple example of a stored procedure, the business object that is constructed from it, and the steps the adapter uses to process the stored procedure business object with an Execute operation.

A simple example of a stored procedure:

```
PROCEDURE testSP(IN int x,INOUT VARCHAR(10) msgSTR, OUT int
status, OUT struct outrec, OUT array retArr)
```

(Procedure returns two resultsets)

For this stored procedure, an example of the business object that is constructed:

```
BOLevel ASI
      SPName=testSP
      ResultSet=true
      MaxNumberOfResultSets=2
      ReturnValue = propName (if function). Will be property name
corresponding to the child business object if returned value is complex type
                           (array/struct/resultset) Defined only if it is a Function

Properties
      x Type=IP
      msgStr Type=IO
      status Type=OP
      outrec Type OP - Child BO for outrec, ASI ChildBOType = struct
      retarr Type OP - n cardinality child BO for retArr, ASI ChildBOType = array
      childBOName1 - Child BO for 1st resultset, ASI ChildBOType = resultset
      childBOName2 - Child BO for 2nd resultset, ASI ChildBOType = resultset
```

To process this stored procedure business object with an Execute operation, the adapter:

1. Constructs the stored procedure call `CALL testSP(x, msgStr, status, outrec)`
2. Sets the input parameters on the callable statement.
3. Executes the callable statement.
4. Obtains the return value (if Function) and sets the value in the appropriate attribute if it is a scalar value, or in a child business object if it is a complex value (such as struct, array).
5. Obtains the first resultset and creates the container for ResultSet1.
6. Obtains the second resultset and creates the container for ResultSet2.
7. Obtains out parameters `msgStr` and `status`, and sets the corresponding attributes on the business object.
8. Obtains out parameter `outrec` and creates the child business object from the data returned in `outrec`. If `outrec` is a nested struct type, then the adapter recursively creates and stores data in the hierarchical child business object.
9. Obtains the out parameter `retArr` and creates an n cardinality child business object from the data returned in `retArr`. If `retArr` is a nested array type, then the adapter recursively creates and stores data in the hierarchical child business object.

Application-specific information

Application-specific information in business object definitions provides the adapter with application-dependent instructions on how to process business objects. The adapter parses the application-specific information from the attributes or verb of a business object or from the business object itself to generate queries for Create, Update, Retrieve, and Delete operations.

The adapter stores some of the business object's application-specific information in cache and uses this information to build queries for all the verbs.

In an extended or modified application-specific business object, the application-specific information in the business object definition must match the syntax that the adapter expects.

Application-specific information at the business-object level

Application-specific information at the business object level is used to specify the name of the corresponding database table and to provide information necessary to perform a physical or logical Delete operation.

At the business object level, application-specific information format consists of eXtensible Markup Language (XML) code that is defined in the jdbcasi.xsd schema definition. The adapter supports the business object level application-specific information listed in the following table.

Table 3. Business object level application-specific information

Application-specific information	**Type**	**Description**	**Bidirectional transformation supported**
MaxNumberOfRetRS	String	The number of result sets that the stored procedure returns	No
ResultSet	Boolean	The value that indicates whether the stored procedure will return a result set	No

Table 3. Business object level application-specific information (continued)

Application-specific information	Type	Description	Bidirectional transformation supported
ReturnValue	String	The name of the corresponding attribute in the stored function business object	No
SelectStatement	String	The select statement	Yes
SPName	String	The name of the stored procedure	Yes
StatusColumnName	String	The name of the database column used to perform logical Delete operations	Yes
StatusValue	String	The value that signifies whether a business object is inactive or deleted	No
TableName	String	The name of the database table associated with the business object	Yes

For example, assume that a Customer business object has the following value specified for its business object application-specific information:

```
<jdbcasi:TableName>customer</jdbcasi:TableName>
<jdbcasi:StatusColumnName>status</jdbcasi:StatusColumnName>
<jdbcasi::StatusValue>deleted</jdbcasi:StatusValue>
```

Assume that the adapter receives a request to delete a customer. Such a request causes the adapter to issue the following SQL statement:

```
UPDATE customer set status = 'deleted' where pkey = . . . .
```

If the StatusColumnName is not included or no value is specified for it, the adapter physically deletes the business object from the database. In other words, if the business object includes the StatusColumnName parameter in its application-specific information, the adapter performs a logical delete operation. If the business object does not include the StatusColumnName parameter in its application-specific information, the adapter performs a physical delete operation.

Both Update and Delete operations can use the value of the StatusColumnName property:

- To logically delete the child data, the adapter uses the value of its StatusColumnName parameter to obtain the name of the status column and the text of the status value. For more information, see “Update operations.”
- When performing a Delete operation, the adapter uses the value of its StatusColumnName parameter to determine whether to physically or logically delete the entire business object. If the StatusColumnName parameter contains a value, the adapter performs a logical delete operation. If the StatusColumnName parameter does not contain a value, the adapter performs a physical delete operation. For more information, see “Delete operations.”

Parameters of the application-specific information that are enabled for use with bidirectional languages are TableName, StatusColumnName, SPName and SelectStatement. The format for these parameters is transformed based on the attributes set for the BiDiEIS, BiDiMetadata, BiDiSkip, and BiDiSpecialFormat properties. For more information on these properties, see “Adapter configuration properties” on page 150.

Verb application-specific information

The adapter uses verb application-specific information to execute operations, such as to retrieve and update information in the database. The adapter updates database tables using SQL queries, stored procedures, or stored functions, as specified in the business objects.

Stored Procedure overview:

A stored procedure is a group of SQL statements that form a logical unit and perform a particular task. A stored procedure encapsulates a set of operations or queries for the adapter to run on an object in a database server.

The adapter can use simple SQL statements for Create, Update, Retrieve, Delete or RetrieveAll operations. The column names for SQL statements are derived from an attribute's application-specific information. The where clause is constructed using key values specified in the business object. Each query spans one table only, unless posted to a view.

The adapter calls stored procedures in the following circumstances:

- Before processing a business object, to perform preparatory operational processes
- After processing a business object, to perform post-operational processes
- To perform a set of operations on a business object, instead of using a simple Create, Update, Delete, Retrieve, or RetrieveAll statement.

When it processes a hierarchical business object, the adapter can use a stored procedure to process the top-level business object or any of its child business objects. However, each business object or array of business objects must have its own stored procedure.

Stored Procedure definition:

Stored procedures are defined at the verb level. Each stored procedure definition consists of the following elements: StoredProcedureType, StoredProcedureName, ResultSet, Parameters, and ReturnValue.

Stored procedure definition elements are described in the following table.

Table 4. Stored procedure definition

Element	Type	Description	Value	Bidirectional transformation supported
StoredProcedure Type	String	Defines the type of stored procedure to be used, and this determines when the stored procedure is called, for example, before processing a business object. **Note:** Stored procedure types associated with RetrieveAll apply to top-level business objects only.	Can be • Before*Verb*SP, • After*Verb*SP, or • *Verb*SP where the verb is either Create, Update, Delete, Retrieve, or RetrieveAll.	No

Table 4. Stored procedure definition (continued)

Element	Type	Description	Value	Bidirectional transformation supported
StoredProcedure Name	String	The name of the stored procedure that is associated with the appropriate StoredProcedureType.		Yes
ResultSet	Boolean	This value determines whether the stored procedure returns a result or not. If the result set is returned, an N-cardinality child for the current business object is created using the values returned in the result set rows.	`true|false`	No
Parameters	String	Defines the list of parameters for stored procedures. **Note:** In the case of Oracle stored procedures, a result set can be returned only as an output parameter. In that case, one of the values in the list of parameters is result set (RS).	A combination of • `IP` for input only, • `OP` for output only, and • `IO` for input and output.	No
ReturnValue	String	A value that indicates it is a function call, not a procedure call, because the value is returned by the function. If the returned value is `RS`, the value is a result set and is used to create the N-cardinality container corresponding to this business object. If the returned value is an attribute, the value is assigned to that particular attribute in the business object. If the attribute is another child business object, the adapter returns an error.	Can be `RS` or a business object attribute.	No

Here is a sample of a stored procedure definition:

```
<jdbcasi:JDBCBusinessObjectTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
          <jdbcasi:TableName>customer</jdbcasi:TableName><jdbcasi:Operation>
             <jdbcasi:Name>Retrieve</jdbcasi:Name>
             <jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>RetrieveSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>retrieve_cust</jdbcasi:StoredProcedureName>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
                  <jdbcasi:Parameters>
                      <jdbcasi:Type>IP</jdbcasi:Type>
                      <jdbcasi:PropertyName>primaryKey</jdbcasi:PropertyName>
```

```
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                          <jdbcasi:Type>OP</jdbcasi:Type>
                           <jdbcasi:PropertyName>custCode</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                          <jdbcasi:Type>OP</jdbcasi:Type>
                           <jdbcasi:PropertyName>firstName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                          <jdbcasi:Type>OP</jdbcasi:Type>
                              <jdbcasi:PropertyName>lastName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    </jdbcasi:StoredProcedures>
                    <jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>AfterRetrieveSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>retrieve_cust</jdbcasi:StoredProcedureName>
                    <jdbcasi:ResultSet>false</jdbcasi:ResultSet>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>IP</jdbcasi:Type>
                           <jdbcasi:PropertyName>primaryKey</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>OP</jdbcasi:Type>
                          <jdbcasi:PropertyName>custCode</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>OP</jdbcasi:Type>
                          <jdbcasi:PropertyName>firstName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>OP</jdbcasi:Type>
                         <jdbcasi:PropertyName>lastName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                 </jdbcasi:StoredProcedures>
             </jdbcasi:Operation>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
```

The Resource adapter property ReturnDummyBOForSP returns output parameters even when the result set is `true` but empty. In the case of RetrieveSP, a result set is returned. If the result set is empty, no business objects are created and there is no way to retrieve the output parameters returned by the procedure call. If ReturnDummyBOForSP is `true`, a dummy business object with values from the output and input/output parameters populated in the corresponding attributes is returned. The default value for this property is `false`.

Stored Functions overview:

Stored functions are similar to stored procedures, except that they always return a value. For some databases, such as Oracle, you can define functions explicitly. For other databases, such as IBM DB2, stored procedures that return a value behave like functions.

A stored procedure call is run as follows:

```
call SPName (<parameter list>)
```

whereas a function call is run as follows:

```
? = call FunctionName (<parameter list>)
```

You specify the attribute that contains the returned value by using the ReturnValue business object application-specific information. For more information on ReturnValue, see "Application-specific information at the business-object level."

Attribute application-specific information

Application-specific information (ASI) for business object attributes differs depending on whether the attribute is a simple attribute, or an attribute that represents a child or an array of child business objects. The application-specific information for an attribute that represents a child differs depending on whether the parent-child relationship is stored in the child or in the parent.

Application-specific information for simple attributes

For simple attributes, the format for application-specific information consists of a number of parameters and their values. The format of attribute application-specific information is shown in the following example section of an .xsd file:

Example .xsd file

```
   <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
   <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
   <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<simpleType>
   <restriction base="string">
   <maxLength value="10"/>
   </restriction>
</simpleType>
</element>
<element name="custCode" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>ccode</jdbcasi:ColumnName>
   <jdbcasi:ForeignKey>custinfoObj/custCode</jdbcasi:ForeignKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="firstName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>fname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="lastName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>lname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
```

The only required parameter for a simple attribute to be processed by the adapter is the column name. For example, this is the format to specify only the column name, where `ccode` stands for customer code:

```
<jdbcasi:ColumnName>ccode</jdbcasi:ColumnName>
```

The attribute application-specific information parameter that is enabled for use with bidirectional languages is ColumnName. The format of this parameter is transformed based on the attributes set for the BiDiEIS, BiDi.Metadata, BiDiSkip and BiDiSpecialFormat properties. For more information on these properties, see ″Adapter configuration properties″ in the ″Reference″ section. For additional information on bidirectional properties, see the bidirectional support general technical paper and the adapter technical paper on the IBM developerWorks(R) Web site.

When you set attribute application-specific information, specify the parameters listed in the table below.

Table 5. Application-specific information for simple attributes

Parameter	**Type**	**Description**	**Default value**	**Bidirectional transformation supported**
BLOB	Boolean	Specifies that the database column that corresponds to this attribute is a BLOB data type. While displaying BLOB data, the adapter only displays the number of bytes as a hexadecimal value. The attribute type is hexBinary.	None	No
ByteArray	Boolean	If `true`, the adapter reads and writes binary data to the database and sends that data as a string to the application server.	`false`	No
ChildBOType	String	Specifies the type of complex parameter as either • Struct, • Array, or • ResultSet	None	No
ChildBOTypeName	String	When the value of the ChildBOType application-specific information is either Struct or Array, this parameter value represents the name of the user-defined type.		No
CLOB	Boolean	Specifies that the database column that corresponds to this attribute is a CLOB data type. Only applicable for String attribute type. **Note:** A CLOB data type is defined as follows: • The CLOB attribute has a String Type whose length is used to define the length of the CLOB. • The CLOB attribute has ASI CLOB=true.	None	No
ColumnName	String	The name of the database column for this attribute.	None	Yes

Table 5. Application-specific information for simple attributes (continued)

Parameter	Type	Description	Default value	Bidirectional transformation supported
DateType	String	Specifies that the corresponding element be treated as Date, Time or TimeStamp type. The DateType parameter has the actual value stored against it, for example, • <DateType>Date</DateType> • <DateType>Time</DateType> • <DateType>TimeStamp</DateType> When setting these values use the following formats: • For Date use yyyy-mm-dd • For Time use hh:mm:ss • For TimeStamp use yyyy-mm-dd hh:mm:ss	None	No
FixedChar	Boolean	Specifies whether the attribute is of fixed length when the columns in the table are of type CHAR, not VARCHAR. For example, if a particular attribute is linked to a column that is of type CHAR, the adapter pads the attribute value with blanks to the maximum length of the attribute when querying the database. This parameter needs to be updated manually in the business object .xsd file. You can edit the file either in text mode or by using the Business Object Editor in WebSphere Integration Developer. Ensure that no validation errors occur in the .xsd after it has been updated. See the code example for this parameter following this table.	`false`	No
ForeignKey	String	The value of this property depends on whether the parent/child relationship is stored in the parent business object or in the child. If it is stored in the parent, set the value to include both the type of the child business object and the name of the attribute in the child to be used as the foreign key *(Child_BO_name/Child_Property_Name)*. If it is stored in the child, set the value to include only the name of the attribute in the parent to be used as the foreign key. If an attribute is not a foreign key, do not include this parameter in the application-specific information.	None	No

Table 5. Application-specific information for simple attributes (continued)

Parameter	Type	Description	Default value	Bidirectional transformation supported
OrderBy	String	If a value is specified for this parameter and the attribute is in a child business object, the adapter uses the value of the attribute in the ORDER BY clause of retrieval queries. The adapter can retrieve child business objects in either ascending order (ASC) or descending order (DESC). If you do not include this parameter in the application-specific information, the adapter does not use this attribute when specifying retrieval order.	None	No
PrimaryKey	Boolean	If the value is `true`, this implies that the column associated with this attribute is a primary key in the corresponding table in the database.	None	No
SPParameterType	String	Specifies the type of stored procedure Parameter: IP (input only), OP (output only), IO (input and output), or RS (result set).	None	No
UniqueIdentifier (UID)	String	The adapter uses this parameter to generate the unique ID for the business object. It supports the generation of sequences and identity columns *(UID=AUTO|Sequence_Name)*. Sequences can be defined for DB2 and Oracle databases only. Identity columns can be defined for DB2 and Microsoft SQL Server. If the attribute does not require a unique ID, do not include this parameter in the application-specific information.	None	No

Example of FixedChar parameter in the business object .xsd file

```
<element name="primaryKey">
<annotation>
<appinfo source="WBI">
        <jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
                <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
                <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
                <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
        </jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<simpleType>
        <restriction base="string">
                <maxLength value="10"/>
        </restriction>
</simpleType>
</element>
```

Application-specific information for attributes of type child business object

Two application-specific information parameters are used for attributes that refer to child business objects (non-simple attributes). When you set this application-specific information, specify the parameters listed in the table below.

Table 6. Application-specific information for attributes of type child business object

Parameter	Type	Description	Default value	Bidirectional transformation supported
KeepRelationship	Boolean	If `true`, this parameter prevents the deletion of a child business object during an Update operation.	None	No
Ownership	Boolean	This parameter specifies that a child business object is owned by the parent. If `true`, then Create, Update, and Delete operations on the child business object are allowed. If `false`, then none of these updates can be applied to the child business object. When its parent is created, the existence of the child is validated to ensure that relationship integrity is maintained in the database.	None	No

Example of OWNERSHIP in the business object .xsd file

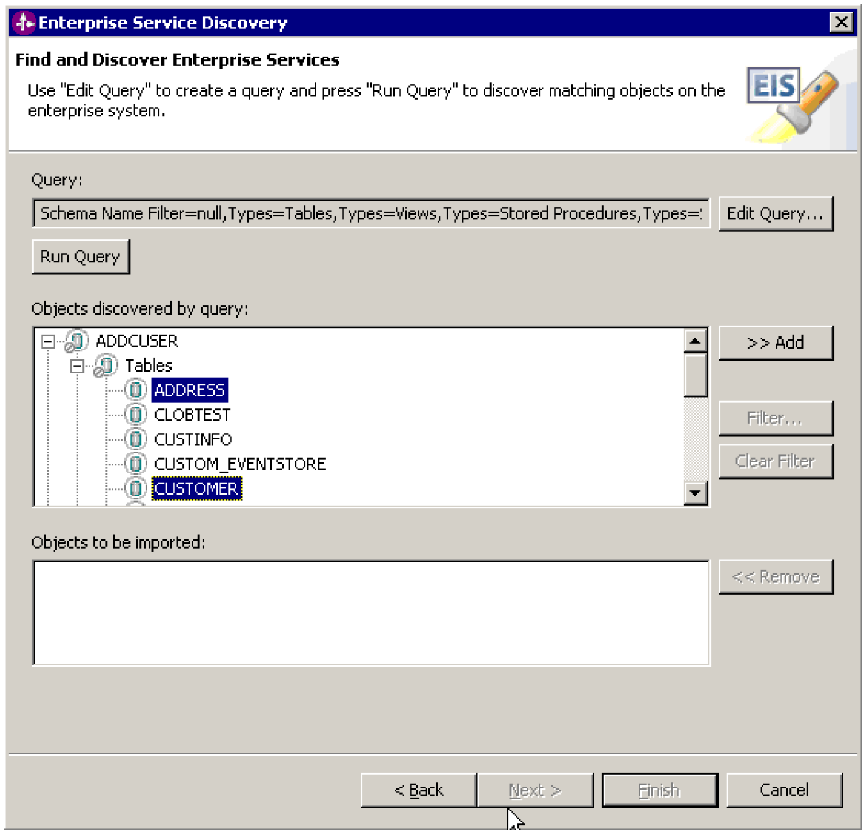
```
<element minOccurs="0" name="addressObj" type="bons0:OutboundRtasserAddress"
 maxOccurs="unbounded">
 <annotation>
   <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
  <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:Ownership>true</jdbcasi:Ownership>
  </jdbcasi:JDBCAttributeTypeMetadata>
   </appinfo>
 </annotation>
</element>


<element minOccurs="0" name="custinfoObj" type="bons1:OutboundRtasserCustinfo"
maxOccurs="1">
 <annotation>
  <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
    <jdbcasi:Ownership>false</jdbcasi:Ownership>
   </jdbcasi:JDBCAttributeTypeMetadata>
  </appinfo>
 </annotation>
</element>
```

Enterprise service discovery

You use the enterprise service discovery wizard in the WebSphere Adapter for JDBC to discover objects in a database and to generate business objects from the selected objects. Enterprise service discovery support also generates the service artifacts that enable the adapter to run as a Service Component Architecture (SCA) component.

You can use the enterprise service discovery wizard to generate business objects from three types of objects in a database. The first type of object includes tables and views, while the second type includes stored procedures and synonyms/nicknames. The third type of object is the query business object, generated from a user-specified select statement.

For the first and second types of objects in the database, enterprise service discovery enables you to discover the objects by generating a list of all the schemas in the database. Within each schema are lists of tables, views, stored procedures, and synonyms/nicknames that you can select, and for which you request the enterprise service discovery wizard to generate the corresponding business objects. The wizard analyzes the metadata of the selected objects and generates attributes in the business objects. The attributes are generated based on the columns of the selected database objects.


Enterprise Service Discovery

Find and Discover Enterprise Services

Use "Edit Query" to create a query and press "Run Query" to discover matching objects on the enterprise system.

Query: Schema Name Filter=null,Types=Tables,Types=Views,Types=Stored Procedures,Types=:

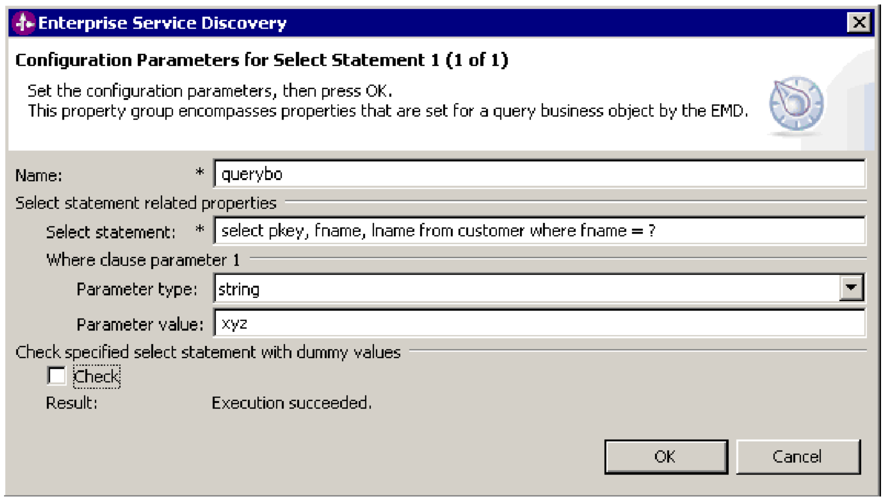
Objects discovered by query:

ADDCUSER
Tables
ADDRESS
CLOBTEST
CUSTINFO
CUSTOM_EVENTSTORE
CUSTOMER

Objects to be imported:

Figure 8. Selecting business objects for import

For the third type of object, the query business object, you specify one select statement. You need to specify the business object name, and the types and dummy values of all the parameters in the where clause. Enterprise service discovery runs this user-specified select statement and analyzes the metadata of the returned result set to obtain the list of columns and their types. After obtaining this information, enterprise service discovery generates one query business object for this user-specified select statement.


Enterprise Service Discovery

Configuration Parameters for Select Statement 1 (1 of 1)

Set the configuration parameters, then press OK.
This property group encompasses properties that are set for a query business object by the EMD.

Name: * querybo

Select statement related properties

Select statement: * select pkey, fname, lname from customer where fname = ?

Where clause parameter 1

Parameter type: string

Parameter value: xyz

Check specified select statement with dummy values

☐ Check

Result: Execution succeeded.

OK Cancel

Figure 9. Configuration Parameters for Select Statement window

To summarize its functions, the enterprise service discovery wizard:

- Generates a business object corresponding to a database object.
- Generates attributes in the business object corresponding to the columns in the database object.
- Sets application-specific information on the business object.
- Provides service descriptions (outbound and inbound) that are used to generate the import, export, and Web Services Description Language (WSDL) files.

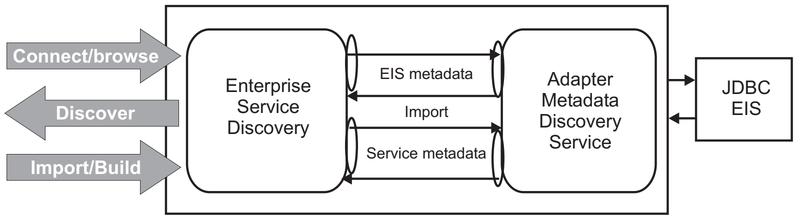


Figure 10. Enterprise Service Discovery functional overview

Discovery of system capabilities

For business object generation based on tables, views, stored procedures, and synonyms/nicknames, enterprise services discovery analyzes the database to identify the schemas. Then the wizard creates a list of all the objects from the database and shows it as a tree structure. You can select any discovered database object to generate a business object. You can also generate query business objects from user-specified select statements.

The schemas are displayed as the top-level nodes in the tree. The schema nodes are not selectable for generation. Under each schema are nodes labeled Tables, Views, Stored Procedures, and Synonyms/Nicknames. These nodes are not selectable for generation. The objects listed under these nodes are the names of the tables, views, stored procedures, and synonyms/nicknames for the particular schema. These nodes are marked as selectable for generation. If no tables, views, stored procedures, or synonyms exist for a particular schema, none are listed.

Filter schemas, node types, and database objects

Before the tree is generated and displayed, you can specify filter properties if you want to narrow the list of schemas displayed in the tree or node types displayed under each schema; otherwise, all schemas and node types are displayed. The SchemaNameFilter property is used to filter schemas, and the Types property is used to filter node types. In addition, you can use the properties DefineASI, QueryBO, and QueryBOCount during object selection. “Filter and node properties” in the “Reference” section describes these properties.

You can select a tables, views, stored procedures, or synonyms/nicknames node, then click **Filter**, and the enterprise service discovery wizard queries for an ObjectNameFilter. You can use the ObjectNameFilter property to filter the list of database objects to display. “Filter and node properties” in the “Reference” section describes the ObjectNameFilter property.

Generate query business objects from user-specified select statements

To generate query business objects from user-specified select statements, select the QueryBO property and set the QueryBOCount to the maximum number of query business objects that can be generated at one time. Then when you run the query to display the tree, you see a new top-level “Query Statements” node along with some schema top-level nodes. Query Statements is not selectable for generation. When you expand this node, then nodes with labels such as “Select Statement 1” and “Select Statement 2” are displayed up to the QueryBOCount number of nodes. These nodes are selectable for generation of query business objects.

Object selection and generation

To generate business objects, you select database object nodes. Then the enterprise service discovery wizard generates business objects for the objects of the selected nodes.

You can select multiple database object nodes. For each table or view type node you select when Define ASI property is set to true, the enterprise service discovery wizard requests the object-level parameters for StatusColumnName and StatusValue application-specific information, and the verb application-specific information association.

For StatusColumnName, you are presented with a list of business object attributes, and you can select one from the list. You need to type a value for the StatusValue. These values are set in the business object level application-specific information.

Associate stored procedures to business objects

In addition, you can choose to associate stored procedures to the business objects. A list of all the supported StoredProcedureType values is presented so you can select the ones you want to configure. For more information, see StoredProcedureType under “Stored Procedure Definition.” After you select some values, the enterprise service discovery wizard displays corresponding verb application-specific information property groups for you to configure.

For example, if you want to configure only the BeforeCreateSP and AfterUpdateSP types, you select these verb application-specific information values for the StoredProcedureType. Then two verb application-specific information property groups are displayed, one for each value.

Each property group contains the properties Schema and StoredProcedure which you use to select the appropriate schema name and stored procedure name. The *Schema property* lists the schemas available in the database. You can filter schemas by using the SPSchemaNameFilter property. You need to select at least one schema from the list of schemas, then the *StoredProcedure property* provides a list of stored procedures available under that schema.

To facilitate selecting stored procedures, for each StoredProcedure property, an SPNameFilter property is provided to narrow the list of stored procedures. Two filter mechanisms are provided: Start With and Wildcard. The supported wildcards are * and ?. When the SPNameFilter property is not set, all stored procedures under the schema are displayed.

Table 7. Stored procedure filter properties

Property	Type	Description
SPSchemaNameFilter	String	Text used to filter schemas. The Schema property must be set. If the SPSchemaNameFilter property is not set, all of the schemas in the database are displayed.
SPNameFilter	String	Text used to filter stored procedures under a schema. StoredProcedure property must be set. Two filter mechanisms are supported: Start With and Wildcard. If the SPNameFilter property is not set, all stored procedures under the schema are displayed.

Table 8. Example of wildcard filter mechanism

Filter criterion	A*B?d
Matched string	**A**dsf**B**n**D**
Matched string	**A**sdfsdfsdf**B**w**d**
Matched string	**A**234dsf**B**6**d**

For each stored procedure assigned, a list of the stored procedure's input and output parameters is presented. Each parameter has a list of the business object's attributes. The parameter's type is listed in the property's description, such as IP, OP, or IO. You may select one business object attribute for each stored procedure parameter. All of the stored procedure types that have a stored procedure assigned are added to the verb application-specific information of the business object, for example, CreateSP and UpdateSP. For more information, see "Verb application-specific information."

You can remove any selected application-specific information from the StoredProcedureType property. All of the corresponding verb application-specific information property groups are also removed.

When you associate a stored procedure with a business object that is generated from a table or view, and if the stored procedure is a function, a value will be returned from this stored procedure. One ReturnValue application-specific information value will be added to the verb application-specific information. The existence of this application-specific information implies that it is a function call and not a procedure call, because a value is being retuned by the function.

If the value of this application-specific information is RS, it implies that the returned value is a result set and that this result set will be used to create the N-cardinality container corresponding to this business object. But if the value of this application-specific information is a business object attribute name, the returned value will be assigned to that particular attribute in the business object.

If the value of this application-specific information is another child business object, the adapter runtime returns an error. In summary, if the returned value is of a simple data type, enterprise service discovery enables you to bind one business object attribute to it, and the value of this application-specific information will be set to the name of that business object attribute. But if the returned value is a result set, enterprise service discovery sets the value of this application-specific information to `RS`.

Note: For an Oracle database, a result set should be returned as an output parameter, not as a returned value. The type of the output parameter is set to `RS` to indicate that this parameter is used to return a result set.

Complex data types

To generate a business object from a stored procedure, which is called a stored procedure business object, you need to set dummy values for the stored procedure input parameters. Then, enterprise service discovery can run this stored procedures to obtain the maximum number of result sets returned and to be able to obtain metadata of these result sets to generate child business objects accordingly.

If the selected stored procedure takes in some complex data types, such as Struct, Array and ResultSet, as input/output parameters or return values, then the wizard queries you for some necessary information. If the parameter is a complex type, you need to select ResultSet/Struct/Array from the property SPComplexParameterType to specify this parameter's data type. In the case of Struct or Array, you also must provide the name of the corresponding user-defined type name in the property SPComplexParameterTypeName.

For example, if you create one Struct object named Struct_TEMP in the database, and you set the type as one input parameter, then you need to set the value of this property to `Struct_TEMP`. Only by obtaining this type name can enterprise service discovery get its metadata to generate a corresponding child business object. If the stored procedure returns ResultSet, you need to set the number of result sets returned from this stored procedure in the property MaxNumberOfResultSets. This value represents the maximum number of returned result sets that will be handled by the runtime.

For stored procedure business objects, enterprise service discovery supports nested structs and arrays, and can support any number of layers of nested hierarchy. Enterprise service discovery can generate corresponding child business objects for all of these nested structs and arrays. The Adapter for JDBC runtime can also support nested structs and arrays.

Table 9. Complex data type properties for stored procedure business objects

Property	Type	Description
SPComplexParameter Type	String	Value can be Struct, Array or ResultSet
SPComplexParameterTypeName	String	The name of the user-defined type. This property is required when the value of SPComplexParameterType is Struct or Array.

Table 9. Complex data type properties for stored procedure business objects (continued)

Property	Type	Description
MaxNumberOfResultSets	Integer	The maximum number of returned result sets to be handled by the Adapter for JDBC runtime. Enterprise service discovery also creates this number of business objects.

QueryBO property and user-specified select statements

If the QueryBO property is selected, the enterprise service discovery wizard asks you for some information. Use the QueryBOName to specify the name for this query business object. Use the SelectStatement property to enter one select statement. When you type the select statement, enterprise service discovery automatically detects the number of parameters in its where clause. When it detects the occurrence of one parameter, it displays one corresponding property group for you to specify the data type and dummy value for this parameter. When you remove one parameter from the select statement's where clause, enterprise service discovery detects this removal, removes all property groups, and recreates necessary property groups for the remaining parameters.

Every such property group includes two properties: WhereParameterType and WhereParameterValue. WhereParameterType has a list of all supported data types. You can select one data type to indicate the actual data type of this parameter in the database. After selecting a data type for this parameter, you can enter one dummy value into the WhereParameterValue property. This value is used to run the select statement.

For example, if the select statement is

```
select * from customer where id=? and age=?
```

enterprise service discovery knows the number of parameters in its where clause is two. Then the wizard displays two property groups for the user to set data types and dummy values for these two parameters. For the first parameter, set the WhereParameterType property to `string` and set the WhereParameterValue to `Mike`. For the second parameter, set the WhereParameterType property to `int` and set the WhereParameterValue to `27`. Enterprise service discovery will run this select statement to get the returned result set.

After obtaining the result set, enterprise service discovery will analyze its metadata to obtain the column name and column type of all columns. For each column of the returned result set, the wizard will generate one corresponding attribute in QueryBO. For each parameter in the where clause, enterprise service discovery will also generate one corresponding attribute in QueryBO. QueryBO generates one attribute named *jdbcwhereclause*. Enterprise service discovery sets the where clause as this attribute's default value. This attribute is used to set one dynamic where clause in runtime to replace the default where clause.

Table 10. Query business object properties

Property	Type	Description
QueryBOName	String	Name of the query business object.
SelectStatement	String	The user-specified select statement.
WhereParameterType	String	The data type of one parameter. Selected from a list of supported data types.

Table 10. Query business object properties (continued)

Property	Type	Description
WhereParameterValue	String	The dummy value of one parameter. Enterprise service discovery uses this value to run the select statement.

All the content data inside business objects has bidirectional transformation support, except for the jdbcwhereclause attribute of QueryBO. Only string constants inside the where clause have bidirectional support.

Important: For correct bidirectional transformation of the jdbcwhereclause attribute, add BiDiContext as attribute level application-specific information and set its special format to SAP WHERE CLAUSE.

Bidirectional support is provided for string constants only in the SelectStatement property, because a SQL statement can be complex and providing bidirectional support could cause its corruption.

After you have selected database objects, you need to set values for the Metadata Selection properties. The enterprise service discovery wizard queries for these properties. For details about these properties, see “Metadata Selection properties” in the “Reference” section.

Data descriptions

Data descriptions are some of the service constructs that the enterprise service discovery process generates to enable the adapter to run as a Service Component Architecture (SCA) component.

The data description includes a definition of the structure and content of adapter business objects that is passed between the client application and the adapter at run time. The data description enables the client application to create the proper data objects for requests, and to interpret the data objects returned as responses. The data description generated from the database components is represented as an XML schema.

- The business object maps to a complex type definition.
- The attributes of the business object map to element type definitions.
- The application-specific information for the business object is contained in annotations at the complex type.
- The application-specific information for each property in the business object is contained in annotations for the element types.

Note: The template for the application-specific properties for the business object and the attribute level are defined in the metadata schema for the JDBC adapter. The name of the schema file is JDBCASI.xsd. The generated schema file has a reference to this template in its annotations.

Business object schema and its application-specific information

The business object schema is built out of database components that you select. Each component translates into a top-level business object.

For those business objects that are based on tables, views, stored procedures, and synonyms/nicknames, the enterprise service discovery wizard generates the name of the business object in the form of *PrefixSchemaNameObjectName*, where

- *Prefix* is the value as specified in the connection property named Prefix. Prefix is not required, and if not specified, no prefix will be added to the business object name.
- *SchemaName* is the name of the schema to which the object belongs.
- *ObjectName* is the name of the table, view, stored procedure, or synonym/nickname.

Globalized characters are supported in the business object name.

For query business objects, the enterprise service discovery wizard generates the same of the business object in the form of *PrefixQueryBOName*, where

- *Prefix* is the value as specified in the connection property named Prefix. Prefix is not required, and if not specified, no prefix will be added to the business object name.
- *QueryBOName* is the value as specified in the configuration property named QueryBOName.

Globalized characters are supported in the business object name.

For the business objects that are based on table or view, the enterprise service discovery wizard sets the TableName application-specific information attribute to a value in the form of *schemaname.tablename*. It sets the business object level application-specific information as listed in the table "Business object application-specific information (ASI) for table or view business objects." The operations you select will be set in the business object. The attributes of the business object are created based on the columns.

Table 11. Business object application-specific information (ASI) for business objects based on tables or views

Business object ASI	**Set by enterprise service discovery wizard**	**Additional information**
TableName	Yes	Set to the actual name of the object
StatusColumnName	Yes	You specify during object selection
StatusValue	Yes	You specify during object selection

For the business objects that are based on stored procedures, the enterprise service discovery wizard sets the business object level application-specific information SPName to a value in the form of *schemaname.spname*. It sets the business object level application-specific information as listed in the table "Business object application-specific information (ASI) for business objects based on stored procedures." The attributes of the business object are created based on the stored procedure input/output parameters. If the stored procedure has one returned value, a corresponding business object attribute is created. If the returned value or any of the input/output parameters are complex data types, the wizard creates child business objects for them.

Enterprise service discovery can support nested structs and arrays. If these child business objects are generated from returned ResultSets, their names are in the form of *PrefixSchemaNameSPNameRetRS#*. For example, if one stored procedure returns two result sets, enterprise service discovery creates two child business objects for them. Their names will be *PrefixSchemaNameSPNameRetRS1* and *PrefixSchemaNameSPNameRetRS2*.

When the child business objects are generated from input/output parameters with a complex data type of ResultSet, Struct, or Array, these child business object names are in the form of *PrefixSchemaNameSPNameParameterName*. For those child business objects that correspond to nested structs and arrays, their business object names are in the form of *PrefixSchemaNameSPNameParameterNameColumnName*.

Table 12. Business object application-specific information (ASI) for business objects based on stored procedures

Business object ASI	**Set by enterprise service discovery wizard**	**Additional information**
SPName	Yes	Set to the actual name of the stored procedure
ResultSet	Yes	Set to `true` if the stored procedure returns one or more ResultSets, otherwise it is set to `false`.
MaxNumofRetRS	Yes	The maximum number of returned result sets that are handled by the adapter runtime.
ReturnValue	Yes	Set to a business object attribute name if the stored procedure has a return value. If the returned value is of simple data type, the attribute is also of simple data type. If the returned value is a ResultSet, this attribute points to a child business object.

For query business objects, one business object level application-specific information named SelectStatement is added. Its value is the complete select statement.

Table 13. Business object application-specific information (ASI) for query business objects

Business object ASI	**Set by enterprise service discovery wizard**	**Additional information**
SelectStatement	Yes	You specify during object selection

The enterprise service discovery wizard also generates business graphs for all business objects, because all are top-level. The name of the business graph will be the business object name followed by "BG." For example, a business object with the name JDBCSchema1Customer, would have a the business graph named JDBCSchema1CustomerBG. The operations set in the business object are also set in the business graph.

When enterprise service discovery generates a stored procedure business object, it creates a child business object if necessary, such as for ResultSet, Struct, and Array. Creating parent-child relationships between table business objects is done manually using the Business Object Editor.

Enterprise service discovery handles business objects based on synonym/nicknames like objects based on tables and views, even when a synonym is of a stored procedure.

Business object attributes and their application-specific information

The attributes of a business object are built from the list of columns in the database object. The enterprise service discovery wizard sets the attribute name to the name of the column. Globalized characters are supported in the attribute names. The adapter adds the attribute name, type, and application-specific information.

The types returned by the JDBC metadata are mapped to the business object attribute types as listed in the table "JDBC metadata column and business object attribute types." Only the JDBC types listed are supported by the adapter. Any columns with types not listed are not added to the business object. An informational message is produced stating, for example, `The column named` *`xxxx`* `in the table named` *`yyyy`* `is not of a supported type and will not be added to the business object.`

Table 14. JDBC metadata column and business object attribute types

JDBC metadata column type	Business object attribute type
BIT	BOOLEAN
CHAR LONGVARCHAR VARCHAR	STRING
INTEGER NUMERIC SMALLINT TINYINT BIGINT	INT
TIME TIMESTAMP DATE	STRING
DECIMAL	STRING
DOUBLE FLOAT	DOUBLE
REAL	FLOAT
BLOB	HEXBINARY
CLOB	STRING
BINARY VARBINARY LONGBINARY	HEXBINARY

The table titled "Attribute information" lists the attribute information set by the enterprise service discovery wizard.

Table 15. Attribute information

Attribute information	Set by enterprise service discovery	Additional information
Cardinality	No	In both single- and multiple-cardinality relationships, the relationship between the parent and child business objects is described by the application-specific information of the key attribute in the business object storing the relationship.
MinOccurs/MaxOccurs	Yes	If the column is not a primary key and is not nullable, the attribute is required, and the values for the attribute will be set to at least 1.
Name	Yes	Name of the attribute. This is enabled for bidirectional languages.
Type	Yes	Set as shown in the table titled "JDBC metadata column and business object attribute types."

The enterprise service discovery wizard sets the attribute application-specific information (ASI) in the business object as shown in the table titled "Attribute application-specific information." For more information, see "Attribute application-specific information."

Table 16. Attribute application-specific information

Attribute ASI	Set by enterprise service discovery	Additional information	Bidirectional transformation supported
BLOB	Yes	Set to `true` if the column data type is BLOB.	No
ByteArray	No	Set to `true` for columns of binary data types. If true, the adapter reads and writes binary data from and to the database. The adapter sets binary data on the business object. The attribute type is hexBinary.	No
ChildBOType	Yes	If the attribute is a complex data type, specify the type as Struct, Array or ResultSet.	No
ChildBOTypeName	Yes	When the value of ChildBOType is Struct or Array, this value represents the name of the user-defined type.	No
CLOB	Yes	Set to `true` if the column data type is CLOB.	No
ColumnName	Yes	Set to the actual name of the column.	Yes
DateType	Yes	Value can be only Date, Time or TimeStamp.	No

Table 16. Attribute application-specific information (continued)

Attribute ASI	Set by enterprise service discovery	Additional information	Bidirectional transformation supported
FixedChar	No	Needs to be updated manually in the business object .xsd file. Use either text mode or the Business Object Editor in WebSphere Integration Developer to edit the file. After updating the file, ensure there are no validation errors. See an example of FixedChar in an .xsd file in the section "Application-specific information for simple attributes."	No
ForeignKey	No	For foreign key attributes, if the parent-child relationship is stored in the parent, the value of this parameter includes both the type of the child business object and the name of the attribute in the child to be used as the foreign key. If the relationship is stored in the child, the value includes only the name of the attribute in the parent to be used as the foreign key.	No
OrderBy	No	If a value is specified, and the attribute is in a child business object, the adapter uses the value of the attribute in the ORDER BY clause of retrieval entries.	No
PrimaryKey	Yes	If the column is a primary key, PrimaryKey is set to `true`.	No
SPParameterType	Yes	If the attribute is associated with a stored procedure parameter, set the value to the actual parameter type, either IP, OP or IO.	No
UID	No	Used when the attribute requires a unique ID for the business object. It supports the generation of sequences and identity columns.	No

Table 17. Attribute application-specific information for attributes of type child business object

Attribute ASI	Set by enterprise service discovery	Additional information	Bidirectional transformation supported
KeepRelationship	No	Prevents deletion of a child business object during an Update operation when the parameter is set to `true`.	No
Ownership	No	Allows Create, Update and Delete operations on the child business object when the parameter is set to `true`.	No

If you choose to add stored procedures to the business objects, the verb application-specific information (ASI) is set as specified in the table titled "Verb application-specific information." For information on valid stored procedure types, see the section "Verb application-specific information."

Table 18. Verb application-specific information

Verb ASI or Stored procedure parameters element	Set by enterprise service discovery	Additional information	Bidirectional transformation supported
Parameters	Yes	Lists the stored procedure parameters.	Yes
PropertyName	Yes	Set to the name of the business object attribute that you select. For more information, see “Object selection and generation.”	Yes
ResultSet	No	If the stored procedure returns a ResultSet, set this parameter to `true` in the business object definition.	No
ReturnValue	Yes	If the stored procedure has a return value, this parameter is set to `RS` or a business object attribute’s name. For more information, see “Object selection and generation.”	No
StoredProcedure	Yes	Set to the stored procedure name.	Yes
StoredProcedure Type	Yes	You choose from a list of types.	No
Type	Yes	Set to the type of the stored procedure parameter, either IP, OP, IO, or RS.	No

To build hierarchical business objects

The enterprise service discovery wizard generates flat table and view business objects. It does not use the foreign key constraints that are defined in the database between different tables to build relationships automatically. These need to be linked manually. You can update the business object definitions either in text mode or by using the Business Object Editor. The enterprise service discovery wizard generates the hierarchy for stored procedure business objects automatically.

An example of the .xsd definition file for single- and multiple-cardinality child business objects is provided here. The element custInfoObj is a single-cardinality child business object, and addressObj is a multiple-cardinality child business object.

Example .xsd file for single- and multiple-cardinality child business objects

```
<element name="addressObj" minOccurs="1" type="Address:Address"
maxOccurs="unbounded">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
"urn:app:jdbc:asi">
                                <pasi:Ownership>true</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
                        <element name="custInfoObj" minOccurs="0" type=
"CustInfo:CustInfo" maxOccurs="1">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
"urn:app:jdbc:asi">
                                <pasi:Ownership>false</pasi:Ownership>
```

```
</pasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
```

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 transformation, which refers 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.

Globalization

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). Components in the WebSphere Business Integration system are written in Java. Therefore, when data is transferred between WebSphere Business 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 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, standards are used to display and process it. WebSphere Process Server and WebSphere Enterprise Service Bus use the Windows standard format, but an enterprise information system exchanging data with WebSphere Process Server or WebSphere Enterprise Service Bus can use a different format. WebSphere Adapters transform bidirectional script data passed between the two systems so that it is accurately processed and displayed on both sides of a transaction.

Bidirectional format

WebSphere Process Server and WebSphere Enterprise Service Bus use the bidirectional format of ILYNN (implicit, left-to-right, on, off, nominal). This is the format used by Windows. If an enterprise information system uses a different format, the adapter converts the format prior to introducing the data to WebSphere Process Server or WebSphere Enterprise Service Bus.

Five attributes comprise bidirectional format. When you set bidirectional properties, you assign values for each of these attributes. The attributes and settings are listed in the following table.

Table 19. Bidirectional format attributes

Letter position	Purpose	Values	Description	Default setting
1	Order schema	I or V	Implicit (Logical) or Visual	I

Table 19. Bidirectional format attributes (continued)

Letter position	Purpose	Values	Description	Default setting
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

The adapter transforms data into a logical, left-to-right format before sending the data to WebSphere Process Server or WebSphere Enterprise Service Bus.

Using bidirectional properties

You can use multiple bidirectional properties to control the transformation of both content data and metadata. You can set special bidirectional properties to exclude either content data or metadata from bidirectional transformation, or to identify data that requires special treatment during a transformation.

The following table describes four types of bidirectional properties.

Table 20. Bidirectional property types

Property type	Data transformations
EIS	Controls the format for content data, or data that is sent by the enterprise information system.
Metadata	Controls the format for metadata, or data that provides information about the content data.
Skip	Identifies content or metadata to exclude from transformation.
Special Format	Identifies certain text, such as file paths or URLs, that require different treatment during the transformation process. Can be set for either content data or metadata.

You can set properties that control bidirectional transformation in three areas.

- **Resource adapter properties:** These properties store default configuration settings, including the TurnBiDiOff property, which controls whether the adapter instance performs bidirectional transformation or not. Use the administrative console of the server to configure these properties.

- **Managed (J2C) connection factory properties:** These properties are used at run time to create an outbound connection instance with an enterprise information system. Once the managed connection factory properties are created, they are stored in the deployment descriptor.
- **Activation specification properties:** These properties hold the inbound event processing configuration information for a message endpoint. Set them as you perform enterprise service discovery, or use the administrative console of the server.

Business object annotations

Some adapters allow you to annotate bidirectional properties within a business object. Do this to add information that specifically controls the transformation of a business object or part of a business object. Use business object editor, a tool within WebSphere Integration Developer, to add annotations at these levels:
- Business object
- Business object application-specific attribute
- Business object attribute
- Business object attribute application-specific attribute

Property scope and lookup mechanism

After you set values for bidirectional properties for an adapter and annotate business objects where appropriate, the adapter performs bidirectional transformations. It does so by using logic that relies on a hierarchical inheritance of property settings and a lookup mechanism.

Properties defined within the resource adapter are at the top of the hierarchy, while those defined within other areas or annotated within a business object are at lower levels of the hierarchy. So for example, if you only set values for EIS-type bidirectional properties for the resource adapter, those values are inherited and used by transformations that require a defined EIS-type bidirectional property whether they arise from an inbound (activation specification) transaction or an outbound (managed connection factory) transaction.

However, if you set values for EIS-type bidirectional properties for both the resource adapter and the activation specification, a transformation arising from an inbound transaction uses the values set for the activation specification.

The processing logic uses a lookup mechanism to search for bidirectional property values to use during a transformation. The lookup mechanism begins its search at the level where the transformation arises and searches upward through the hierarchy for defined values of the appropriate property type. It uses the first valid value it finds. It searches the hierarchy from child to parent only; siblings are not considered in the search.

Chapter 5. Planning for adapter implementation

Before you install the adapter, find out how to improve the performance and availability of the adapter by using a clustered server environment. You can also review the roadmap for installing, configuring, and deploying the adapter to learn at a high level what to do to accomplish these tasks.

WebSphere Adapters in clustered environments

You can improve adapter performance and availability by deploying the WebSphere adapter enterprise archive (EAR) module to a clustered server environment. The adapter instance within the EAR module is replicated across federated servers.

WebSphere Process Server and WebSphere Application Server Network 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 container to activate the adapter instance. The JCA container provides a runtime environment for adapter instances. For more information about clustered environments, see http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_cluster_v61.html.

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

High availability for inbound operations

Inbound operations are based on events triggered as a result of updates to data in the enterprise information system (EIS) application. The adapter is configured to detect updates through event listeners or by polling an event table. The adapter then publishes the event to its endpoint.

In a clustered environment, two or more adapter instances might detect the same event. This scenario raises the possibility of duplicate event processing or data infidelity. For example, if two adapter instances are simultaneously polling the same event table with the same event type filter, one may alter data that the other adapter instance depends on, or it might fail. There is a parallel risk for event-listening adapter architectures in a clustered environment.

To avoid this condition, the HAManager for the inbound adapter instances enforces a singleton behavior. Even though all the adapter instances are started, only one of the instances detects and publishes an event to the endpoint for each type of EIS application.

When you deploy an adapter module to a cluster, the JCA container checks the enableHASupport property of the ResourceAdapter bean. If the value for the enableHASupport property is true, the JCA container registers all of the adapter instances with HAManager with a policy 1 of N. This policy means that only one of the clustered servers starts event polling (or listening) for this adapter instance. 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.

High availability for outbound operations

In clustered environments, multiple adapter instances are available to perform outbound requests. Accordingly, if your environment has multiple applications that interact with the same WebSphere adapter for outbound requests, then you might improve performance by deploying the adapter module to a clustered environment.

WebSphere Application Server Network Deployment has a workload management capability that distributes the outbound processing among the adapter instances. As a result, outbound operations in a clustered environment are similar to those in a single server environment: one adapter instance processes only one outbound request at a time. For more information on workload management, see http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html.

Note: Adapter instances are replicated in a clustered server environment. When the enableHASupport property is set to true, which is the default setting, only one of the replicated adapter instances actively polls for events while other instances are in standby mode. If the enableHASupport property is set to false, all of the adapter instances replicated on cluster members actively poll for events. This may result in event duplication. Do not change the value of enableHASupport to false for single server environments. For information on changing the value of this property, see the Resource adapter properties section in this documentation. To determine whether adapter replication is supported in a clustered environment, see the software and hardware requirements section of this documentation.

Mainframe data access

The Adapter for JDBC supports mainframe data access through use of the IBM WebSphere Information Integrator Classic Federation for z/OS®. This product provides Web and distributed applications with read/write connectivity to mainframe databases.

The product delivers high-performance SQL-driven access and federation of mainframe data sources. You can use the desktop with your choice of Internet tools and applications to access mainframe, mission-critical information transparently.

If you have problems using the Adapter for JDBC with IBM WebSphere Information Integrator Classic Federation on the IBM z/OS platform, refer to technotes describing configuration requirements on the IBM product support Web pages at http://www-306.ibm.com/software/integration/wbiadapters/support.

Roadmap for installing, configuring, and deploying the adapter

Before you can use the adapter in a runtime environment, you must install, configure, and deploy it. Understanding these tasks at a high level helps you perform the steps that are needed to accomplish each task.

After successfully installing the WebSphere Adapter, you configure it using WebSphere Integration Developer. You then deploy it as an enterprise archive (EAR) file to WebSphere Process Server or WebSphere Enterprise Service Bus. The following figure illustrates this flow of tasks, and the steps that follow the figure describe each task at a high-level. For detailed instructions on installing, see *Installing IBM WebSphere Adapters*. For information about configuring and deploying the adapter, see the adapter documentation.

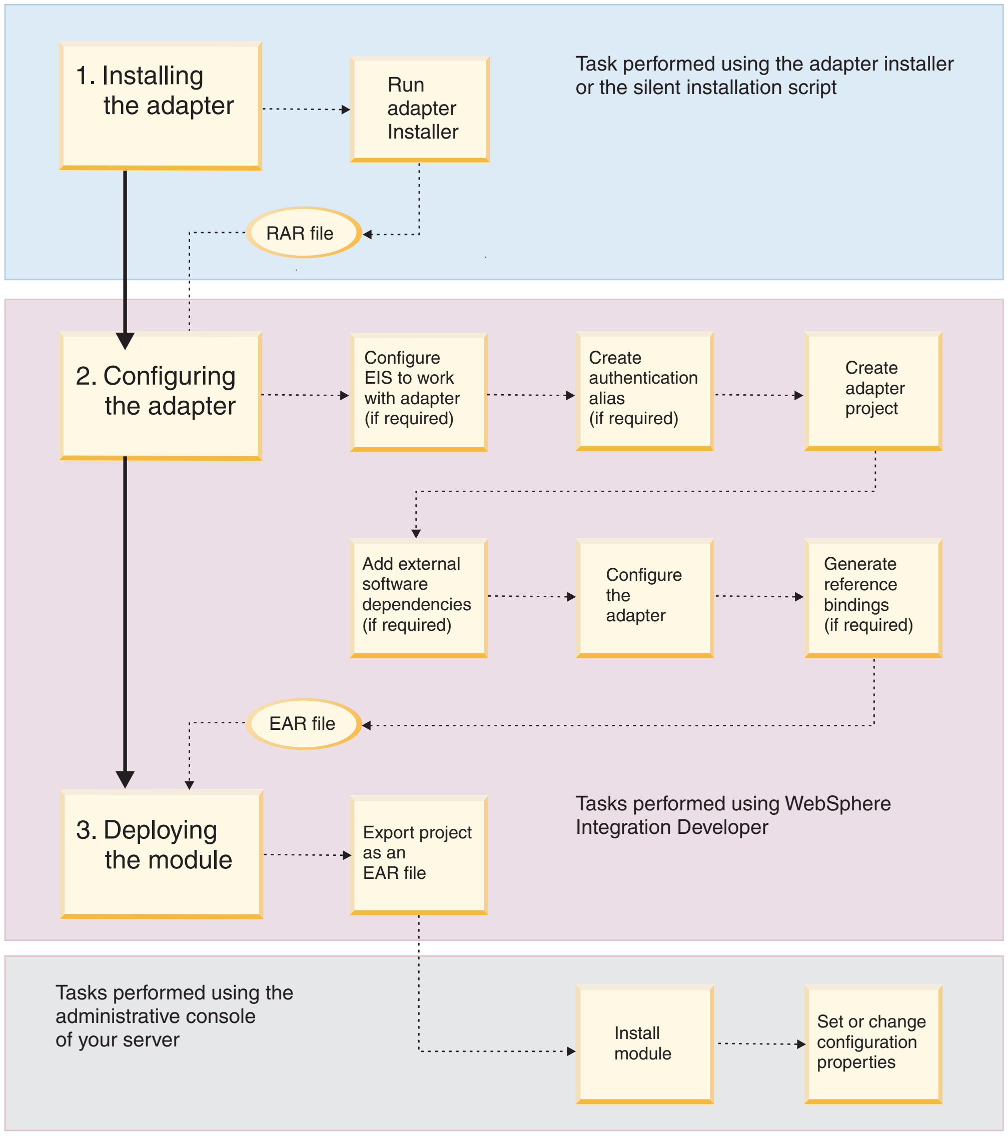

1. Installing the adapter
Run adapter Installer
Task performed using the adapter installer or the silent installation script
RAR file
2. Configuring the adapter
Configure EIS to work with adapter (if required)
Create authentication alias (if required)
Create adapter project
Add external software dependencies (if required)
Configure the adapter
Generate reference bindings (if required)
EAR file
3. Deploying the module
Export project as an EAR file
Tasks performed using WebSphere Integration Developer
Tasks performed using the administrative console of your server
Install module
Set or change configuration properties

Figure 11. Roadmap for installing, configuring, and deploying the adapter

1. **Installing the adapter**
   a. Use the installer (a graphical user interface) or a script that runs a silent installation. Either method installs a resource adapter archive (RAR) file on your workstation. You use this RAR file to configure the adapter.
2. **Configuring the adapter**
   a. (If required) Configure the enterprise information system (EIS) to work with your adapter. You perform this step from within the EIS application.

b. (If required) Create an authentication alias to access the application.

c. Create an adapter project in WebSphere Integration Developer (J2EE Perspective) by importing the adapter RAR file.

d. (If required) Using WebSphere Integration Developer, add any external dependencies required by your adapter to the adapter project. These dependencies are also required as part of the bundled EAR file, which is exported when you deploy the adapter.

e. To configure the adapter, run the enterprise service discovery wizard from the Business Integration Perspective of WebSphere Integration Developer. The enterprise service discovery wizard generates business integration components and allows you to enter all the information necessary to configure the adapter for the first time. The output from the enterprise service discovery tool is saved to a business integration module project, which contains the business object, or objects, and the import or export file.

f. (If required) Use WebSphere Integration Developer to generate reference bindings for the component created by the enterprise service discovery wizard.

3. **Deploying the module**

   a. From the J2EE perspective in WebSphere Integration Developer, export a business integration module project as an EAR file.

   b. Install the module on WebSphere Process Server or WebSphere Enterprise Service Bus.

   c. (If required) In the server administrative console, set (or change) the following properties:

      - Resource adapter properties
      - Managed (J2C) connection factory properties
      - Activation specification properties for the EIS

Chapter 6. Installing the adapter

To install the adapter, you must check system prerequisites, perform migration steps, and perform the installation steps common to all adapters. Finally, you perform additional installation steps specific to the WebSphere Adapter for JDBC.

Installation prerequisites

Before installing the Adapter for JDBC, you must verify that your environment meets all of the necessary hardware and software requirements. These requirements fall into two categories: supported platforms for running the adapter installer, and hardware and software requirements for configuring, deploying, and running the adapter.

Supported platforms for running the adapter installer

The supported platforms for running the adapter installer are located in the ″Installing″ section of Installing IBM WebSphere Adapters.

Hardware and software requirements for configuring, deploying, and running the adapter

The hardware and software requirements for configuring, deploying, and running the adapter are located at the following Web site: IBM WebSphere Adapters and IBM WebSphere Business Integration Adapters: software requirements. From the IBM WebSphere Adapters list, select the link for the Adapter for JDBC, Version 6.0.2.

Additional JAR files

If you are using WebSphere Integration Developer version 6.0.1.1 or earlier, you must manually add three additional JAR files to the class path of the connector project. For more information on how to do this, see ″Adding JAR files to WebSphere Integration Developer versions 6.0.1.1 and earlier″ in the “Reference” section.

Migrating to version 6.0.2

To migrate from WebSphere Adapter for JDBC version 6.0.0.x, first review the information on backward compatibility and follow the instructions to replace the RAR file with the version 6.0.2 RAR file. Then edit some Activation specification properties and add the xid column to the event table.

Backward compatibility

This version of the Adapter for JDBC runtime is backward compatible with version 6.0.0.x. Consequently, enterprise archive (EAR) files that are generated with the 6.0.0.x version of the adapter work as-is with the Adapter for JDBC version 6.0.2 runtime.

Migration considerations

Previous versions of the Adapter for JDBC resource adapter archive (RAR) file must be replaced with the Adapter for JDBC version 6.0.2 RAR file.

To replace the RAR file, first import the project interchange (PI) file for the deployed enterprise archive (EAR) file into WebSphere Integration Developer.

Then remove the old RAR file, and import the latest adapter RAR file.

Performing the migration

Migration to Adapter for JDBC version 6.0.2 requires that you edit the Inbound Service description by updating some of the Activation specification properties. You also need to set properties if you want to do event filtering. You must change the event table to add the transaction id (xid) column.

Before you begin

Replace the previous version of the resource adapter archive (RAR) file with the latest adapter RAR file.

How to perform this task

1. Edit the Inbound Service description by updating properties

   After you have imported the project interchange (PI) file generated using version 6.0.0.x of the Adapter for JDBC, use the assembly editor to edit the Inbound service description by updating only the Activation specification properties listed in the following table.

Table 21. Activation specification properties to update during migration

Property	Type	Description	Default value	Globalized
AssuredOnceDelivery	Boolean	If this property is set to `true`, an xid value is set for each event in the event store. Each event is then delivered to its corresponding endpoint and subsequently deleted from the event table.	true	No
CustomDeleteQuery	String	The custom delete query that is run after each event is processed. This is enabled for use with bidirectional languages.	None	Yes
CustomEventQuery	String	The SQL query, stored procedure, or stored function for custom event processing. This query is run during each poll cycle when the EventQueryType is set to Dynamic. This is enabled for use with bidirectional languages.	None	Yes
CustomUpdateQuery	String	The custom update query that is run after each event is processed so that the same event does not get picked up for processing in the subsequent event cycle. This is enabled for use with bidirectional languages.	None	Yes
DataSource JNDIName	String	Name used by the adapter to establish the connection to the database. If UserName and Password are also set, they are also used when establishing the connection. If not, just the DataSourceJNDIName property is used.	None	Yes

Table 21. Activation specification properties to update during migration (continued)

Property	Type	Description	Default value	Globalized
EventFilterType	String	The adapter can filter the events to be processed by business object type. EventFilterType has a comma delimited list of business object types, and only the types specified in the property are picked up for processing. If no value is specified for the property, no filter is applied, and all the events are picked up for processing.	null	Yes
EventQueryType	String	Determines whether to use the standard event store or custom query. The valid values are `Standard`, for standard event store, and `Dynamic`, for custom event processing.	None	No
FilterFutureEvents	Boolean	If this property is set to `true`, the adapter filters events based on timestamp. The adapter compares the system time in each poll cycle to the timestamp on each event. If an event is set to occur in the future, it is not picked up for processing until that time.	false	No
SPAfterPoll	String	Any stored procedure that you want to be run after each poll cycle. It takes one input parameter for PollQuantity. This is enabled for use with bidirectional languages.	None	Yes
SPBeforePoll	String	Any stored procedure that you want to be run before the actual poll query is called. It takes one input parameter for PollQuantity. This is enabled for use with bidirectional languages.	None	Yes

2. **Optional:** Set properties if you want to filter events

   You can use two Activation specification properties to enable event filtering: FilterFutureEvents (for event filtering by timestamp) and EventFilterType (for event filtering by business object type).

   FilterFutureEvents is set to false by default. Change the setting if you want to filter events by timestamp.

   To filter by business object type, set a comma-delimited list of business object types in the EventFilterType property, otherwise, leave it blank.
3. Add the xid column to the event table

   Note: Before you alter the event table, ensure that no events are in the Event Distribution Table (EDT). The Adapter for JDBC, version 6.0.2 does not use the EDT, but uses a different mechanism to provide assured once delivery of events. Also ensure no events are being added to the event table while the migration is in progress.

   Use the following sample SQL statement to alter the event table:

   ```
   ALTER TABLE <tablename> ADD COLUMN xid VARCHAR(255);
   ```

Performing the installation

The basic steps for installing the adapter are the same for all WebSphere Adapters. You can install the adapter either by using a graphical user interface or by performing a silent installation. After performing the common installation steps, you must also perform the installation step specific to the WebSphere Adapter for JDBC.

Before you begin

Review the installation prerequisites.

How to perform this task

1. Install the adapter using the basic installation instructions, which are common to all adapters. These steps are located in the Installing section of Installing IBM WebSphere Adapters.
2. Perform the following step that is specific to the WebSphere Adapter for JDBC.
   a. Set up the JDBC driver so that the JDBC driver JAR file is in the class path.

      JDBC enterprise service discovery exists within a project in the enterprise service discovery wizard. The JDBC driver JAR file needs to be in the class path of the JDBC project in this tool to enable you to run the JDBC enterprise service discovery process.

Result

The resource adapter archive (RAR) file is copied to the workstation where the adapter is installed. If you accepted the default installation location, the RAR file is placed in the following directory: C:\Program Files\IBM\ResourceAdapters\JDBC\adapter\jdbc\deploy\CWYBC_JDBC.rar.

What to do next

Configure the adapter.

Installed file structure

After you install the adapter, you can view the installed files and directories, all of which have the installation directory as their root.

For example, if the installation directory for the adapter is c:\WebSphereBI, then the CWYBC_JDBC.rar file has the following absolute path: `c:\WebSphereBI\adapter\jdbc\deploy\CWYBC_JDBC.rar`

The resource adapter archive (RAR) file contains both the adapter and the enterprise service discovery wizard files.

UNIX® and Windows platforms share the same installed directory and file structure, with the only difference being the directory path designation (forward slash / for UNIX, backslash \ for Windows).

The following table lists the UNIX/Linux directories and files for the WebSphere Adapter for JDBC. Directories and files are grouped into categories.

Table 22. Directory and file structure for UNIX/Linux

File and directory category	Directories and files
Adapter for JDBC RAR file	adapter/jdbc/deploy/CWYBC_JDBC.rar
IBM Support Assistant plug-in zip file	adapter/jdbc/ISAPlugin/ com.ibm.esupport.client.SS6FE6_RAJDBC.zip
IBM Tivoli® License Manager (ITLM) file	adapter/jdbc/5724L77E060002.sys
Message files	adapter/jdbc/messages/CWYBC_JDBC_messages.tar
Message files for Foundation Classses	adapter/jdbc/messages/ CWYBS_AdapterFoundation_messages.tar
Sample tutorial Project Interchange file	adapter/jdbc/samples/referencefiles/Tutorial1.zip
Sample scripts for IBM DB2	adapter/jdbc/samples/scripts/scripts_db2.sql
Sample scripts for Oracle Sample scripts for Microsoft SQL	adapter/jdbc/samples/scripts_oracle.sql adapter/jdbc/samples/scripts_mssql.sql

The following table lists the Windows directories and files for the WebSphere Adapter for JDBC. Directories and files are grouped into categories.

Table 23. Directory and file structure for Windows

File and directory category	Directories and files
Adapter for JDBC RAR file	adapter\jdbc\deploy\CWYBC_JDBC.rar
IBM Support Assistant plug-in zip file	adapter\jdbc\ISAPlugin\ com.ibm.esupport.client.SS6FE6_RAJDBC.zip
IBM Tivoli License Manager (ITLM) file	adapter\jdbc\5724L77E060002.sys
Message files	adapter\jdbc\messages\CWYBC_JDBC_messages.zip
Message files for Foundation Classses	adapter\\jdbc\messages\ CWYBS_AdapterFoundation_messages.zip
Sample tutorial Project Interchange file	adapter\jdbc\samples\referencefiles\Tutorial1.zip
Sample scripts for IBM DB2	adapter\jdbc\samples\scripts\scripts_db2.sql
Sample scripts for Oracle Sample scripts for Microsoft SQL	adapter\jdbc\samples\scripts_oracle.sql adapter\jdbc\samples\scripts_mssql.sql

Uninstalling the adapter

The steps for uninstalling the adapter are the same for all WebSphere Adapters. You can uninstall the adapter either by using a graphical user interface or by performing a silent uninstallation.

About this task

Uninstalling the adapter may be a required task for troubleshooting an installation problem. The steps for uninstalling the adapter are located in the ″Uninstalling″ section of Installing WebSphere Adapters.

Note: If you need to uninstall an adapter that is already deployed, refer to the ″Additional adapter-related information you might need″ section of “Related product information” on page 161.

Chapter 7. Configuring the adapter for deployment

To configure WebSphere Adapter for JDBC so that it can be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, use WebSphere Integration Developer to create an adapter project, add required files to the project, and specify the business objects you want to discover and the system on which you want to discover them.

Configuring the EIS to work with the adapter

You need to set up the event store in the database before doing inbound event processing. You can set triggers on user tables as needed.

How to perform this task

1. Set up the event store. Before inbound processing can occur, you need to set up the event store in the database. Sample scripts are provided to set up the event store for the IBM DB2, Oracle, or Microsoft SQLServer database, as follows:
   - `WBIA_JDBC_EventStore_DB2.sql`
   - `WBIA_JDBC_EventStore_Oracle.sql`
   - `WBIA_JDBC_EventStore_MSSQL.sql`
2. **Optional:** Set up triggers on user tables. As needed, set up triggers on user tables so that changes to the user tables can automatically generate events that are stored in the event store.

Creating the authentication alias

Create the authentication alias by using the administrative console on the server. From the administrative console, configure the global security and set the password for the authentication alias, which is used to process outbound requests.

About this task

Before you configure the adapter, you must create an authentication alias on the server. To create the authentication alias, use the following procedure.

How to perform this task

1. On the administrative console ″Welcome page,″ click **Security** → **Global security**.
2. Under the Authentication heading, click **JAAS Configuration** → **J2C Authentication data**.
3. Click **New**.
4. Type the required information in the Alias, User ID, Password, and Description fields.

   Note: The user ID and password that you type will be used to establish a connection to the database for outbound processing.
5. Click **OK**.
6. Click **Save**, and then click **Save** again.

Creating the adapter project in WebSphere Integration Developer

To begin the process of creating and deploying a module, you create an adapter project. The adapter project contains the adapter itself plus other related artifacts. You create the project by importing the RAR file, which was copied to your local file system during installation, into WebSphere Integration Developer.

Before you begin

Make sure you have installed the Adapter for JDBC and that you have created an authentication alias.

About this task

Create an adapter project (called a *connector project* in WebSphere Integration Developer) to contain the adapter (which you import from the adapter installation directory) as well as artifacts related to it. All projects are self-contained; they do not refer to objects outside of the project.

To create an adapter project, use the following procedure.

How to perform this task

1. If WebSphere Integration Developer is not currently running, start it now.
   a. Click **Start** → **Programs** → **IBM WebSphere** → **Integration Developer V6.0.2** → **WebSphere Integration Developer V6.0.2**.
   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, close the Welcome page.
2. Switch to the J2EE perspective:
   a. Click **Window** → **Open Perspective** → **Other**.
   b. Click **J2EE**.
      If **J2EE** is not displayed in the Select Perspective window, select the **Show all** check box, click **J2EE**, and click **OK**.
   c. If you see the Confirm Enablement window, select **Always enable capabilities and don't ask me again**.
   d. Click **OK**.
3. Import the RAR file by right-clicking **Connector Projects** and clicking **Import** → **RAR file**.

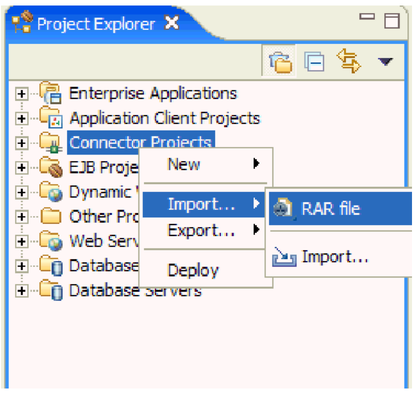
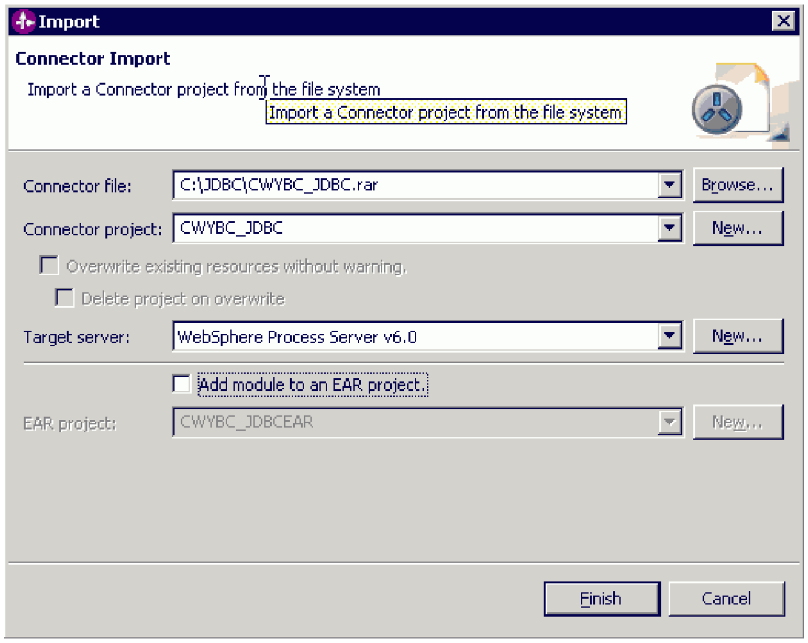


Figure 12. Importing the RAR file

4. From the Connector Import window, click **Browse** and navigate to the directory in which the Adapter for JDBC was installed.
5. Click **CWYBC_JDBC.rar**.

   The connector project has the same name as the RAR file.

   If a project named CWYBC_JDBC.rar already exists in this workspace, the name in the **Connector project** field has a number appended to it (for example, CWYBC_JDBC1).
6. **Optional:** In the **Connector project** field, type another name for the project or accept the default value.
7. **Optional:** In the **Target server** field, select the server to which the adapter will be deployed or accept the default value.
8. Clear the **Add module to an EAR project** check box.



Import

Connector Import

Import a Connector project from the file system

Connector file: C:\JDBC\CWYBC_JDBC.rar

Connector project: CWYBC_JDBC

☐ Overwrite existing resources without warning.

☐ Delete project on overwrite

Target server: WebSphere Process Server v6.0

☐ Add module to an EAR project.

EAR project: CWYBC_JDBCEAR

Figure 13. Clearing the Add module to an EAR project check box

   Notice that the EAR project field becomes unavailable after you remove the check mark.
9. Click **Finish**.

10. Add the JDBC driver to the connector project.

    You must add the JDBC driver to the connector project so that it becomes part of the EAR file that you deploy to the application server. You can do this either after you import the RAR file, or after you install the EAR on the application server.

Option	Description
To add the JDBC driver after you import the RAR file	Add the JAR file to the appropriate folder in the workspace. For example, the JAR file could be added to this location: `c:\workspace\CWYBC_JDBC\connectorModule`
To add the JDBC driver after you install the EAR on the application server	After you install the application, add the JAR file to the RAR subdirectory of the installedApps directory of WebSphere Process Server or WebSphere Enterprise Service Bus. For details on installing the application on the application server, see "Deploying the module."

Result

A new J2EE Connector project is created. To see its contents, expand the project in Project Explorer. For example, if the connector project is named CWYBC_JDBC, expand **CWYBC_JDBC**.

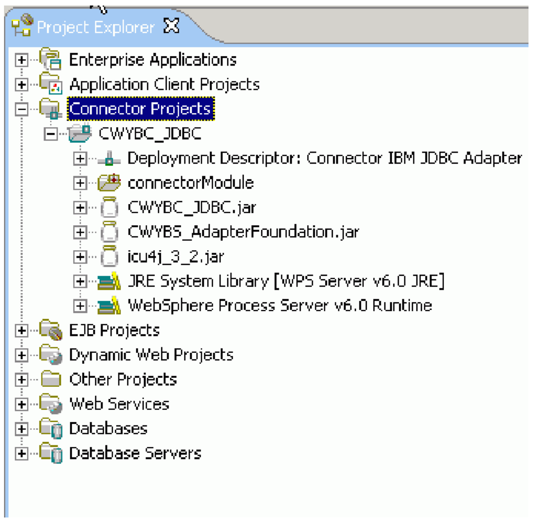


Figure 14. The connector project in the Project Explorer window

What to do next

Add the required external dependencies to the project.

Adding external software dependencies

After you create the project in WebSphere Integration Developer, you need to add the database driver to your adapter project. To accomplish this, add the JDBC driver to the Java build path. Use the external JAR file that is provided with your database.

How to perform this task

1. In WebSphere Integration Developer in the J2EE perspective, expand **Connector Projects**. Right-click the adapter project **CWYBC_JDBC** and select **Properties**.

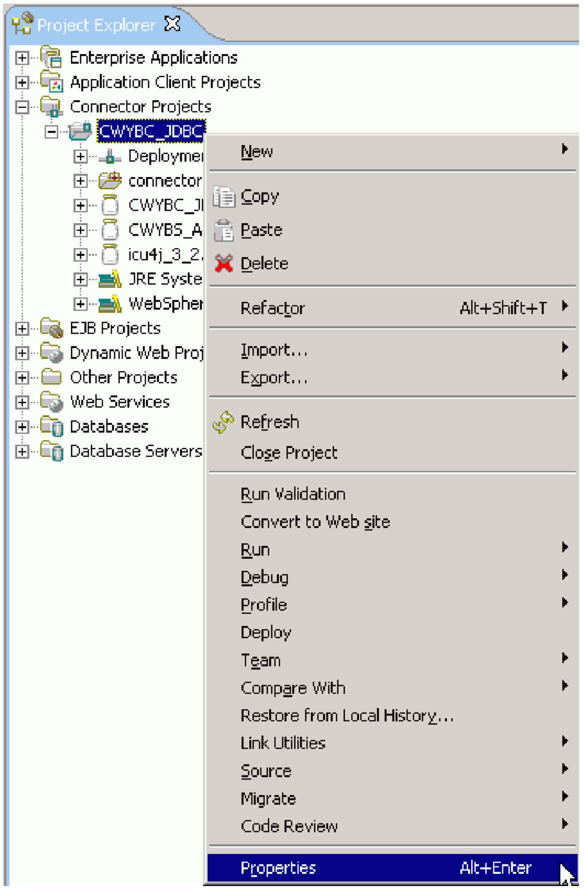


Figure 15. Selecting Properties option for your project

2. In the navigation sidebar, click **Java Build Path**. Select the **Libraries** tab and click **Add External JARs**.
3. In the JAR Selection window, navigate to the directory on your local file system that contains the JDBC driver JAR file. Click **Open**.

   The JDBC driver JAR file is added to the Java build path.
4. Click **OK**.

Result

Your adapter project contains a reference to the JDBC driver, which you can view in the Project Explorer window of WebSphere Integration Developer.

Configuring the adapter for outbound processing

To configure WebSphere Adapter for JDBC for outbound processing, use the enterprise service discovery wizard in WebSphere Integration Developer to set the connection properties for enterprise service discovery, select business objects or services that are in the enterprise information system, and generate business object definitions and related artifacts for outbound processing.

Generating business objects using enterprise service discovery

To generate business objects, you first set connection properties. Then you run a query for database objects so that you can select objects needed for the service description. You configure the selected objects, and save the artifacts and property values in a new business integration module

Setting connection properties for enterprise service discovery

After you have created the adapter project and added the database driver, you need to start the enterprise service discovery wizard for the Adapter for JDBC. You use the wizard to set the values of the connection properties for your database instance.

About this task

The enterprise service discovery process requires these properties to connect to the database for discovery and for creating the service description.

How to perform this task

1. Start enterprise service discovery.
   a. In WebSphere Integration Developer, switch to the **Business Integration** perspective by selecting **Window** → **Open Perspective** → **Other**. Click **Business Integration (default)** and click **OK**.
   b. Select **File** and right-click **New > Enterprise Service Discovery**.

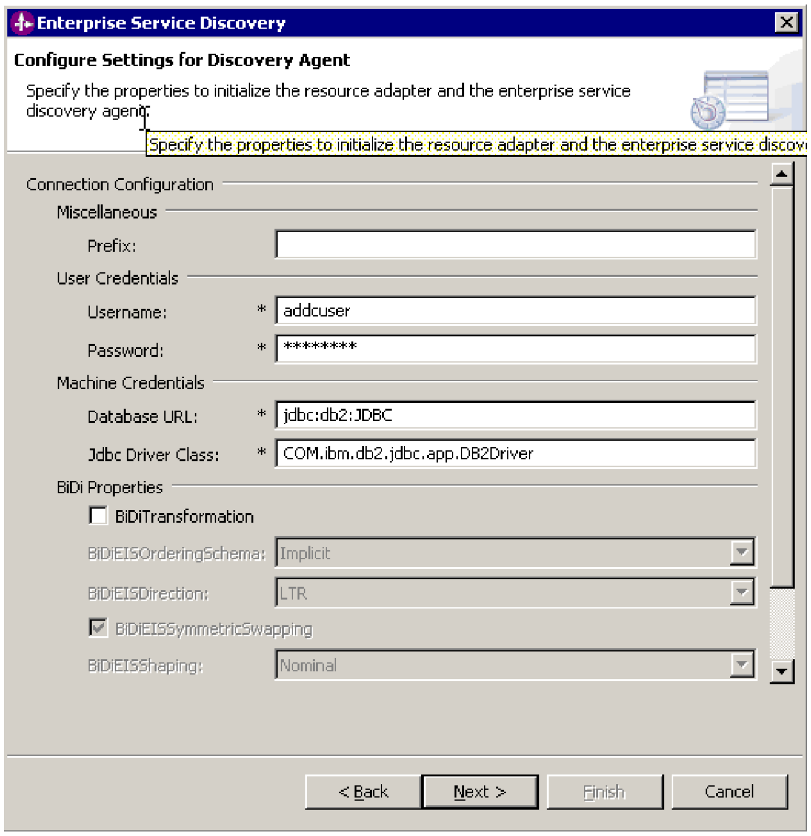
      If **Enterprise Service Discovery** is not displayed, click **New** → **Other**, expand **Business Integration**, and click **Enterprise Service Discovery**. Then click **Next**.
   c. In the Enterprise Service Discovery window, select the option for your adapter and click **Next**.

      If you had not already imported the RAR file, you could click **Import Resource Adapter** in this window to import it.
2. Set values of connection properties.

   You need to set the values of the Enterprise service discovery connection properties used to connect to the target EIS instance for discovery and for creating the service description. To enable bidirectional script data processing, you need to activate bidirectional transformation and set values for the bidirectional properties.
   a. In the Configure Settings for Discovery Agent window, type the values for the **Connection Configuration** properties.

      The required values are marked with asterisks. Type your Username, Password, Database URL, and JDBC Driver Class. Check your driver documentation for the appropriate values for Database URL and JDBC Driver Class.
   b. To activate bidirectional capability, select the check box next to **BiDiTransformation**. Then set the values for the bidirectional properties.

   For details on these properties, see "Enterprise service discovery connection properties" in the "Reference" section.



Enterprise Service Discovery

Configure Settings for Discovery Agent

Specify the properties to initialize the resource adapter and the enterprise service discovery agent.

Connection Configuration

Miscellaneous

Prefix:

User Credentials

Username: * addcuser

Password: * ********

Machine Credentials

Database URL: * jdbc:db2:JDBC

Jdbc Driver Class: * COM.ibm.db2.jdbc.app.DB2Driver

BiDi Properties

☐ BiDiTransformation

BiDiEISOrderingSchema: Implicit

BiDiEISDirection: LTR

☑ BiDiEISSymmetricSwapping

BiDiEISShaping: Nominal

Figure 16. Configure settings for Discovery Agent window

3. Select logging options.

   Logging options are used to set up logging and tracing for the enterprise service discovery process only. However, the logging and tracing levels are the same as for the adapter. For details on logging and tracing levels for the adapter, see "Configuring troubleshooting tools."

   a. At the bottom of the Configure Settings for Discovery Agent window, click **Show Advanced**. The button changes to **Hide Advanced**.

   b. Under **Logging options**, type or browse for the output location of the log file. Select the logging and tracing levels for enterprise service discovery.

   c. Click **Next**.

Result

The discovery service uses the connection properties to prepare a metadata tree that is displayed for object selection and navigation.

Selecting business objects and services

After configuring the connection properties, run a query for database objects. Browse the metadata tree structure to understand the structure of objects in the enterprise information system (EIS), and make selections of objects needed for the service description.

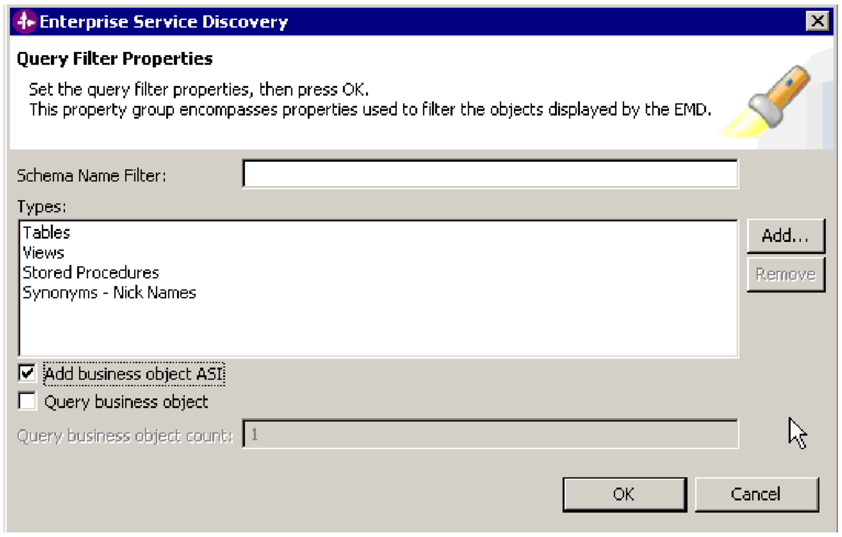
Before you begin

Before running the query, you can specify Filter properties if you want to narrow the list of schemas, nodes, or objects displayed in the tree structure.

How to perform this task

1. Specify Filter properties.
   a. In the Find and Discover Enterprise Services window, click **Edit Query**. In the Query Filter Properties window, type text in the **Schema Name Filter** property field. The schemas that start with the string you specify are displayed. Select the schemas that you want to use.
   b. The **Types** property field lists the entries: tables, views, stored procedures, and synonyms/nicknames. You can add or remove nodes from that list.
   c. In the Query Filter Properties window, you can check **Add Business Object ASI** and click **OK**.

      Then whenever you add an object when you run the metadata query in Step 2, the Configuration Parameters for *(name of object)* window is displayed for entering application-specific information.


Enterprise Service Discovery

Query Filter Properties

Set the query filter properties, then press OK.
This property group encompasses properties used to filter the objects displayed by the EMD.

Schema Name Filter:

Types:
Tables
Views
Stored Procedures
Synonyms - Nick Names

[✓] Add business object ASI
[ ] Query business object
Query business object count: 1

Figure 17. Query Filter Properties window

2. Run the metadata query. In the Find and Discover Enterprise Services window click **Run Query**.

   The objects discovered by the query are displayed in the top pane. The tables, views, stored procedures, and synonyms/nicknames are sorted by schema name.
3. Filter objects.

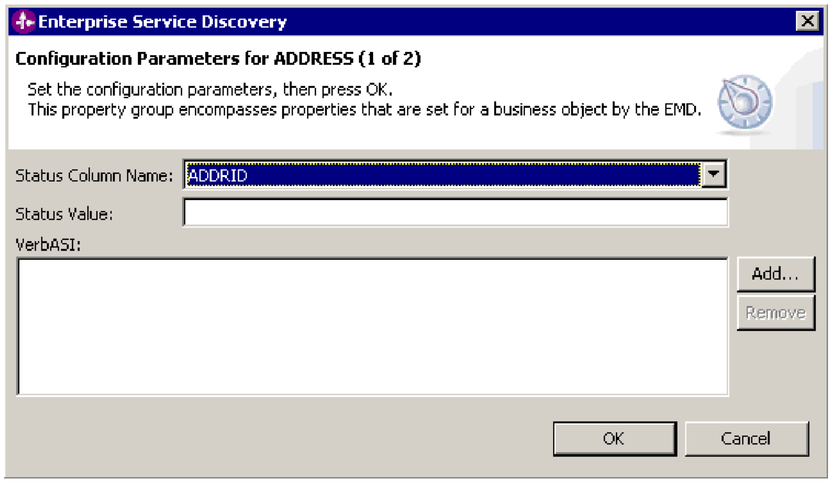
   You can filter objects similar to the way you filtered schemas.
   a. Select a tables, views, stored procedures, or synonyms/nicknames node. Click **Filter**. The enterprise service discovery wizard queries for an **Object Name Filter** to filter the list of database objects to display for that node.
   b. Type in text, and only those database objects that start with the specified string are displayed.
4. Select an object for import by highlighting an object and clicking **Add**. The selected objects appear in the bottom pane.

   Note: If you need to remove a selected object, highlight it and click **Remove**.
5. Add business object ASI.

   If you checked **Add Business Object ASI** in the Query Filter Properties window, then whenever you add an object, a window called Configuration

Parameters for *(name of object)* appears for entering application-specific information. This window is shown here with verb application-specific information parameters for the **ADDRESS** object.

The **Status Column Name** field is used for logical deletes, and **Status Value** specifies what values to update in the column corresponding to the attribute chosen for Status Column Name.


Enterprise Service Discovery

Configuration Parameters for ADDRESS (1 of 2)

Set the configuration parameters, then press OK.
This property group encompasses properties that are set for a business object by the EMD.

Status Column Name: ADDRID

Status Value:

VerbASI:

Figure 18. Configuration Parameters for ADDRESS window

a. Click **Add**.

b. A list of stored procedures is displayed. Select a stored procedure to associate the stored procedure application-specific information with the ADDRESS object. Click **OK**.

   For example, if your stored procedure creates a record in a table, enter the details for the **CreateSP** configuration parameter.

c. Type the values for the **Schema** name and **Stored Procedure** name. When you enter a stored procedure name, the **Stored Procedure Parameters** fields are displayed. For each parameter, select a value from the list of business object attributes to associate with the parameter.

d. Click **OK** when you are finished entering ASI.

e. The Find and Discover Enterprise Services window shows the selected objects in the **Objects to be imported** pane. Click **Next**.

If you select more than one object to add, the Configuration Parameters for *(name of object)* window for the first object appears. After you enter the ASI and click **OK**, the window for the next object appears, and so on. For information on the object-level, verb, and attribute application-specific information, see "Application-specific information."

Result

The selected objects have been imported for use with the adapter.

Configuring the selected objects

After you have selected database objects, you need to specify values for the Metadata selection properties for the import file.

Before you begin

For more information, see the “Metadata selection properties” in the “Reference” section.

How to perform this task

1. Specify Name Space.

   The default value is displayed in the Configure objects window. The default is the Name Space for the metadata schema, JDBCASI.xsd. Name Space is prepended to the business object name to keep the business object schemas logically separated.
2. Select **Outbound** for **Service Type**.

   Query business objects and stored procedure business objects support only service type outbound.
3. Select operations.
   a. In the Configure objects window, the **Operations** field lists the operations that the adapter supports for the Service Type you selected. To add to the list of operations, click the **Add** button.

      For example, you might remove an operation and then decide you want to include it.
   b. At the Add window, select from the list of operations and click **OK**.

      The specified operations are set for all business objects being generated.
4. Set Max Records by typing the maximum number of records to retrieve for a RetrieveAll operation. The default value is 100.
5. Specify BO Location by typing the path to the location where the generated .xsd files are to be stored.
6. When you are finished, click **Next**.

Generating artifacts

After you specify Metadata selection properties, you configure properties that the adapter uses to set up a communication channel to a specific database. These properties include Resource adapter and Managed (J2C) connection factory properties. You also need to create a new business integration module where all the artifacts and property values can be saved.

How to perform this task

1. Specify new module name.
   - In the Generate artifacts window, which is shown below, if a new module name appears in the **Module** field, you don’t need to take any action.
   - Otherwise, you need to click **New**. If the New Integration Project window is displayed, select **Create a module project** and click **Next.** In the New Module window, enter the module name, and click **Finish**.
2. **Optional:** In the Generate Artifacts window, you can specify a folder for the service description by typing or browsing for a folder within the new module where the service description will be saved. If you do not specify a folder name, the artifacts (the import and WSDL files) are stored in the root folder of the module, that is, the folder with the module name.
3. The import file name appears in the **Name** field. You could add a comment in the **Description** field.
4. Click **Edit Operation names** to review the default names of all the operations for the business objects you are creating.

   A default name comprises the operation name combined with the business object name.

5. Select the **Deploy connector with module** check box.

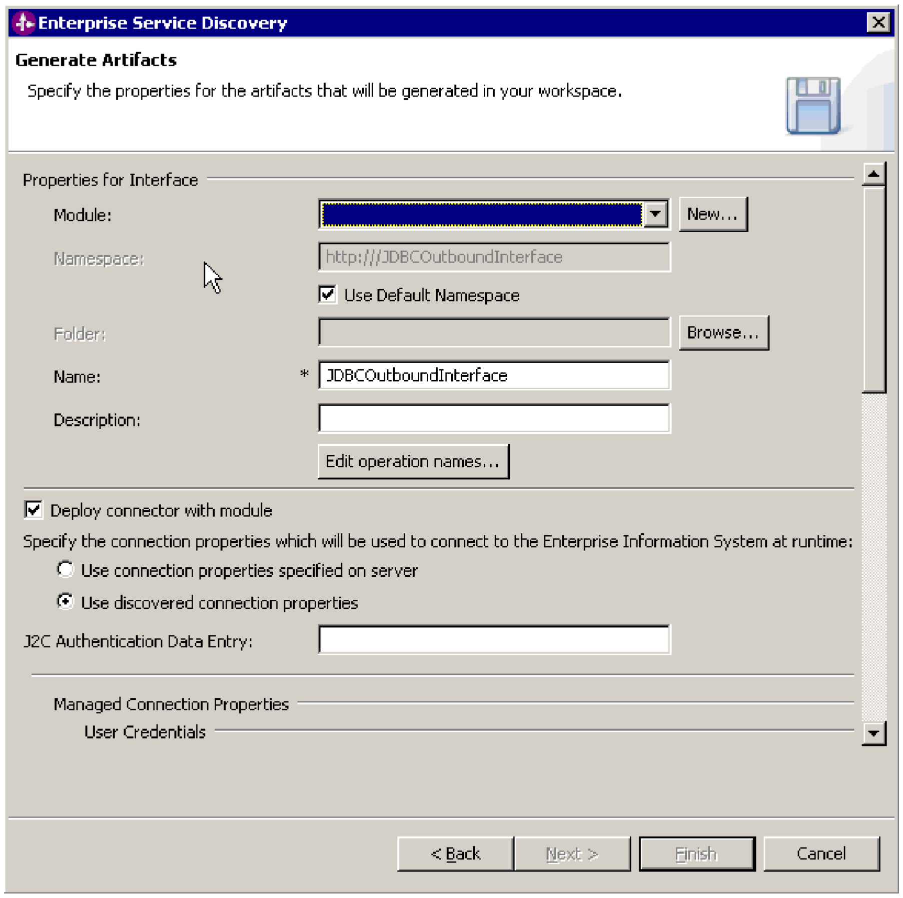
   Note: By checking **Deploy connector with module** you ensure that the adapter project RAR file is included in the EAR file that you deploy to the application server.
6. Specify configuration property values that will be used to connect to the EIS at run time.
   - Select **Use discovered connection properties** to set property values.
   - In the future after you deploy the project and you need to create new business objects, you can select **Use connection properties specified on server**. This selection indicates that you want to use the properties already on the application server.
7. Set property values. Required property fields are marked with an asterisk.

   For outbound processing, the property fields are displayed for the Managed (J2C) connection factory and Resource adapter properties.

   If you activated bidirectional transformation, the bidirectional property fields are displayed for outbound processing.

   For details on these configuration properties, see “Adapter configuration properties” in the “Reference” section.
8. Specify in the **J2C Authentication Data Entry** field the authentication alias that you set up on the application server.

   The name is case-sensitive. For more details, see “Creating the authentication alias.”
9. Type the connection properties **Username**, **Password**, **Database URL**, and **JDBC Driver Class** that you set up after starting the enterprise service discovery wizard.


Enterprise Service Discovery

Generate Artifacts

Specify the properties for the artifacts that will be generated in your workspace.

Properties for Interface

Module:

Namespace: http:///JDBCOutboundInterface

[✓] Use Default Namespace

Folder:

Name: * JDBCOutboundInterface

Description:

[✓] Deploy connector with module

Specify the connection properties which will be used to connect to the Enterprise Information System at runtime:

( ) Use connection properties specified on server

(●) Use discovered connection properties

J2C Authentication Data Entry:

Managed Connection Properties

User Credentials

Figure 19. Generate artifacts window

10. When you are finished setting required properties, click **Finish**.

Generating reference bindings

Reference bindings are used by other WebSphere Business Integration Service Component Architecture (SCA) components to access the adapter. You create a reference to the adapter from the project module to link the adapter to other server processes.

About this task

Reference binding generation is required in a testing environment only. It is not necessary when deploying the adapter in a production environment.

How to perform this task

1. Create SCA component
   a. Go to the **Business Integration** perspective of WebSphere Integration Developer. In the **Business Integration** tab, right-click the **JDBCEMD** module, and select **Open With > Assembly Editor**.
   b. The Assembly Diagram window is displayed with the module's Import component in view. To create a new component, click the second icon (Import) in the left-hand (vertical) frame of the window.
   c. Click the bottom icon in the new menu of icons (it has hover-help that reads **Standalone References**). The cursor changes to the placement icon.

   d. Click the palette to add the new component to the Assembly Diagram window.
2. Create a standalone reference
   a. Click and drag the module's Import component to the new component. This draws a wire from the Import component to the new component.
   b. In the Add Wire window, click **OK**. Another Add Wire window is displayed, asking if you want to convert the WSDL interface to Java. Click **No**.
   c. The new **Standalone Reference** component displays in the Assembly Diagram window with a wire that connects it to the module's Import component.

      Click **File > Save** to save the assembly diagram.

      For more information, refer to the WebSphere Integration Developer user guide at http://www.ibm.com/software/integration/wid.

Configuring the adapter for inbound processing

To configure WebSphere Adapter for JDBC for inbound processing, use the enterprise service discovery wizard in WebSphere Integration Developer to set the connection properties for the adapter, select business objects or services that are in the enterprise information system, and generate business object definitions and related artifacts for inbound processing.

Generating business objects using enterprise service discovery

To generate business objects, you first set connection properties. Then you run a query for database objects so that you can select objects needed for the service description. You configure the selected objects, and save the artifacts and property values in a new business integration module

Setting connection properties for enterprise service discovery

After you have created the adapter project and added the database driver, you need to start the enterprise service discovery wizard for the Adapter for JDBC. You use the wizard to set the values of the connection properties for your database instance.

About this task

The enterprise service discovery process requires these properties to connect to the database for discovery and for creating the service description.

How to perform this task

1. Start enterprise service discovery.
   a. In WebSphere Integration Developer, switch to the **Business Integration** perspective by selecting **Window** → **Open Perspective** → **Other**. Click **Business Integration (default)** and click **OK**.
   b. Select **File** and right-click **New > Enterprise Service Discovery**.

      If **Enterprise Service Discovery** is not displayed, click **New** → **Other**, expand **Business Integration**, and click **Enterprise Service Discovery**. Then click **Next**.
   c. In the Enterprise Service Discovery window, select the option for your adapter and click **Next**.

If you had not already imported the RAR file, you could click **Import Resource Adapter** in this window to import it.

2. Set values of connection properties.

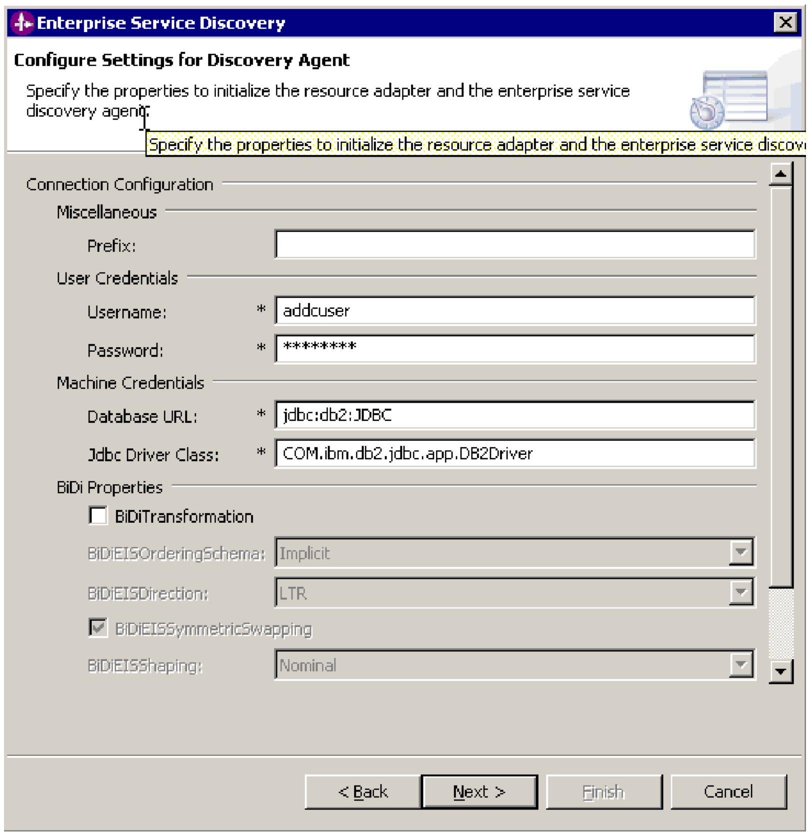
   You need to set the values of the Enterprise service discovery connection properties used to connect to the target EIS instance for discovery and for creating the service description. To enable bidirectional script data processing, you need to activate bidirectional transformation and set values for the bidirectional properties.

   a. In the Configure Settings for Discovery Agent window, type the values for the **Connection Configuration** properties.

      The required values are marked with asterisks. Type your Username, Password, Database URL, and JDBC Driver Class. Check your driver documentation for the appropriate values for Database URL and JDBC Driver Class.

   b. To activate bidirectional capability, select the check box next to **BiDiTransformation**. Then set the values for the bidirectional properties.

   For details on these properties, see "Enterprise service discovery connection properties" in the "Reference" section.



Enterprise Service Discovery

Configure Settings for Discovery Agent

Specify the properties to initialize the resource adapter and the enterprise service discovery agent.

Connection Configuration

Miscellaneous

Prefix:

User Credentials

Username: * addcuser

Password: * ********

Machine Credentials

Database URL: * jdbc:db2:JDBC

Jdbc Driver Class: * COM.ibm.db2.jdbc.app.DB2Driver

BiDi Properties

☐ BiDiTransformation

BiDiEISOrderingSchema: Implicit

BiDiEISDirection: LTR

☑ BiDiEISSymmetricSwapping

BiDiEISShaping: Nominal

Specify the properties to initialize the resource adapter and the enterprise service discov
< Back
Next >
Finish
Cancel

Figure 20. Configure settings for Discovery Agent window

3. Select logging options.

   Logging options are used to set up logging and tracing for the enterprise service discovery process only. However, the logging and tracing levels are the same as for the adapter. For details on logging and tracing levels for the adapter, see "Configuring troubleshooting tools."

a. At the bottom of the Configure Settings for Discovery Agent window, click **Show Advanced**. The button changes to **Hide Advanced**.
b. Under **Logging options**, type or browse for the output location of the log file. Select the logging and tracing levels for enterprise service discovery.
c. Click **Next**.

Result

The discovery service uses the connection properties to prepare a metadata tree that is displayed for object selection and navigation.

Selecting business objects and services

After configuring the connection properties, run a query for database objects. Browse the metadata tree structure to understand the structure of objects in the enterprise information system (EIS), and make selections of objects needed for the service description.

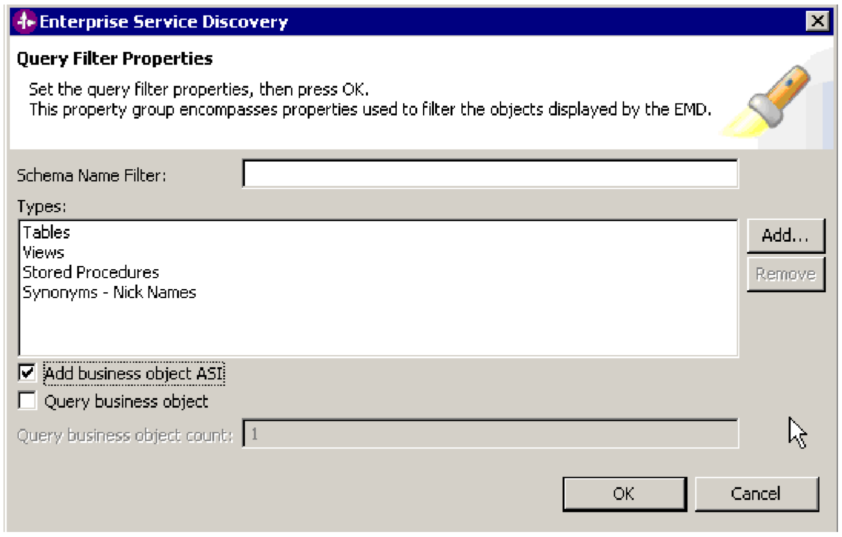
Before you begin

Before running the query, you can specify Filter properties if you want to narrow the list of schemas, nodes, or objects displayed in the tree structure.

How to perform this task

1. Specify Filter properties.
   a. In the Find and Discover Enterprise Services window, click **Edit Query**. In the Query Filter Properties window, type text in the **Schema Name Filter** property field. The schemas that start with the string you specify are displayed. Select the schemas that you want to use.
   b. The **Types** property field lists the entries: tables, views, stored procedures, and synonyms/nicknames. You can add or remove nodes from that list.
   c. In the Query Filter Properties window, you can check **Add Business Object ASI** and click **OK**.

      Then whenever you add an object when you run the metadata query in Step 2, the Configuration Parameters for *(name of object)* window is displayed for entering application-specific information.


Enterprise Service Discovery

Query Filter Properties

Set the query filter properties, then press OK.
This property group encompasses properties used to filter the objects displayed by the EMD.

Schema Name Filter:

Types:

Tables
Views
Stored Procedures
Synonyms - Nick Names

☑ Add business object ASI

☐ Query business object

Query business object count: 1

Figure 21. Query Filter Properties window

2. Run the metadata query. In the Find and Discover Enterprise Services window click **Run Query**.

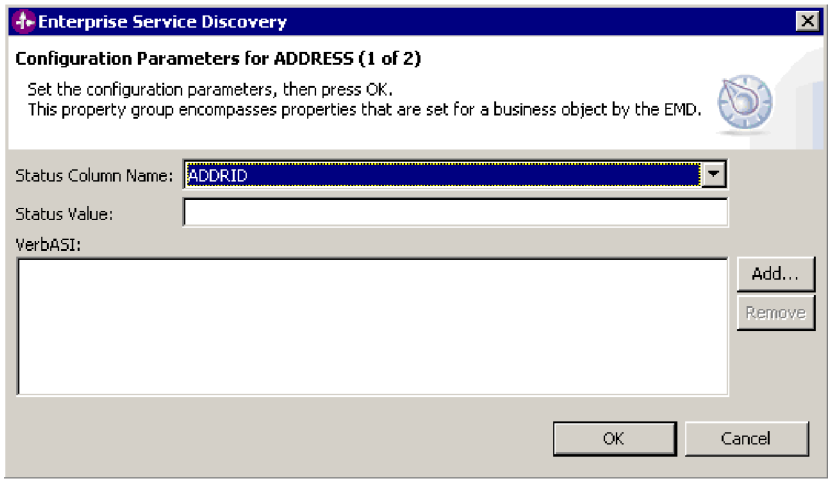
   The objects discovered by the query are displayed in the top pane. The tables, views, stored procedures, and synonyms/nicknames are sorted by schema name.
3. Filter objects.

   You can filter objects similar to the way you filtered schemas.
   a. Select a tables, views, stored procedures, or synonyms/nicknames node. Click **Filter**. The enterprise service discovery wizard queries for an **Object Name Filter** to filter the list of database objects to display for that node.
   b. Type in text, and only those database objects that start with the specified string are displayed.
4. Select an object by highlighting an object and clicking **Add**. The selected objects appear in the bottom pane.

   Note: If you need to remove a selected object, highlight it and click **Remove**.
5. Add business object ASI.

   If you checked **Add Business Object ASI** in the Query Filter Properties window, then whenever you add an object, a window called Configuration Parameters for *(name of object)* is displayed for entering application-specific information. This window is shown here with verb application-specific information parameters for the **ADDRESS** object.

   The **Status Column Name** field is used for logical deletes, and **Status Value** specifies what values to update in the column corresponding to the attribute chosen for Status Column Name.


Enterprise Service Discovery

Configuration Parameters for ADDRESS (1 of 2)

Set the configuration parameters, then press OK.
This property group encompasses properties that are set for a business object by the EMD.

Status Column Name: ADDRID

Status Value:

VerbASI:

Figure 22. Configuration Parameters for ADDRESS window

a. Click **Add**.
b. A list of stored procedures is displayed. Select a stored procedure to associate the stored procedure application-specific information with the ADDRESS object. Click **OK**.

   For example, if your stored procedure creates a record in a table, enter the details for the **CreateSP** configuration parameter.
c. Type the values for the **Schema** name and **StoredProcedure** name. When you enter a stored procedure name, the **StoredProcedureParameters** fields are displayed. For each parameter, select a value from the list of business object attributes to associate with the parameter.
d. Click **OK** when you are finished entering ASI.
e. The Find and Discover Enterprise Services window shows the selected objects in the **Objects to be imported** pane. Click **Next**.

If you select more than one object to add, the Configuration Parameters for *(name of object)* window for the first object is displayed. After you enter the ASI and click **OK**, the window for the next object is displayed, and so on. For information on the object-level, verb, and attribute application-specific information, see “Application-specific information.”

Configuring the selected objects

After you have selected database objects, you need to specify values for the Metadata selection properties for the export file.

Before you begin

For details on the Metadata selection properties, see the “Reference” section.

How to perform this task

1. Specify Name Space.

   The default value is displayed in the Configure objects window. The default is the Name Space for the metadata schema, JDBCASI.xsd. Name Space is prepended to the business object name to keep the business object schemas logically separated.
2. Select **Inbound** for the **Service Type**.

   Query business objects and stored procedure business objects support only service type outbound.

3. Select operations.
   a. In the Configure objects window, the **Operations** field lists the operations that the adapter supports for the Service Type you selected. To add to the list of operations, click the **Add** button.

      For example, you might remove an operation and then decide you want to include it.
   b. At the Add window, select from the list of operations and click **OK**.

      The specified operations are set for all business objects being generated.
4. Do not set a value for Max Records; it is not applicable to inbound processing.
5. Specify BO Location by typing the path to the location where the generated .xsd files are to be stored.
6. When you are finished, click **Next**.

Generating artifacts

After you specify Metadata selection properties, you configure properties that the adapter uses to set up a communication channel to a specific database. These properties include Resource adapter and Activation specification properties. You also need to create a new business integration module where all the artifacts and property values can be saved.

How to perform this task

1. Specify new module name.
   - In the Generate artifacts window, which is shown below, if a new module name appears in the **Module** field, you don't need to take any action.
   - Otherwise, you need to click **New**. If the New Integration Project window is displayed, select **Create a module project** and click **Next**. In the New Module window, enter the module name and click **Finish**.
2. **Optional:** In the Generate Artifacts window, specify a folder for the service description by typing or browsing for a folder within the new module where the service description will be saved. If you do not specify a folder name, the artifacts (the export and WSDL files) are stored in the root folder of the module, that is, the folder with the module name.
3. The export file name appears in the **Name** field. You could add a comment in the **Description** field.
4. Click **Edit Operation names** to review the default names of all the operations for the business objects you are creating.

   The default names comprise the operation name combined with the business object name.
5. Select the **Deploy connector with module** check box.

   Note: By checking **Deploy connector with module** you ensure that the adapter project RAR file is included in the EAR file that you deploy to the application server.
6. Specify configuration property values that will be used to connect to the EIS at run time.
   - Select **Use discovered connection properties** to set property values.
   - In the future after you deploy the project and you need to create new business objects, you can select **Use connection properties specified on server**. This selection indicates that you want to use the properties already on the application server.
7. Set property values. Required property fields are marked with an asterisk.

For inbound processing, the property fields are displayed for the Activation specification and Resource adapter properties.

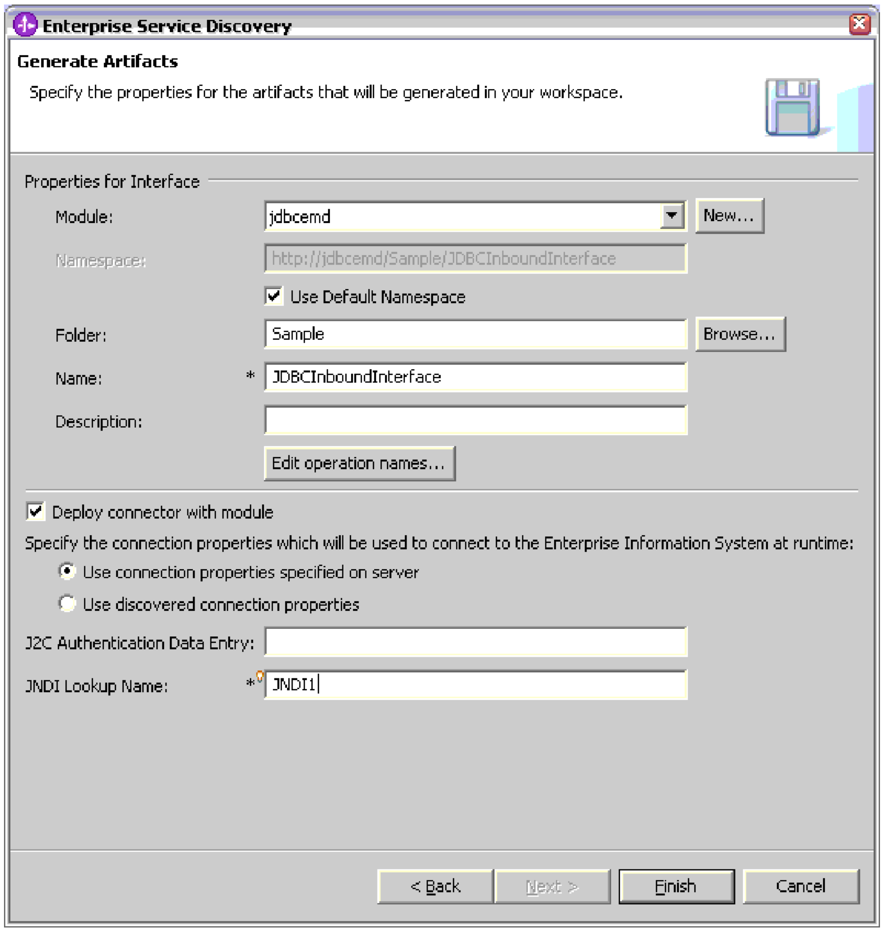
If you activated bidirectional transformation, the bidirectional property fields are displayed for inbound processing.

For details on these configuration properties, see “Adapter configuration properties” in the “Reference” section.

8. Specify in the **J2C Authentication Data Entry** field the authentication alias that you set up on the application server.

   For more details, see “Creating the authentication alias”.

9. Specify in the **JNDI Lookup Name** field the name of the connection factory that will be used on the application server.


Enterprise Service Discovery

Generate Artifacts

Specify the properties for the artifacts that will be generated in your workspace.

Properties for Interface

Module: jdbcemd

Namespace: http://jdbcemd/Sample/JDBCInboundInterface

[✓] Use Default Namespace

Folder: Sample

Name: * JDBCInboundInterface

Description:

[✓] Deploy connector with module

Specify the connection properties which will be used to connect to the Enterprise Information System at runtime:

(●) Use connection properties specified on server

( ) Use discovered connection properties

J2C Authentication Data Entry:

JNDI Lookup Name: * JNDI1

Figure 23. Generate artifacts window

Generating reference bindings

Reference bindings are used by other WebSphere Business Integration Service Component Architecture (SCA) components to access the adapter. You create a reference to the adapter from the project module to link the adapter to other server processes.

About this task

Reference binding generation is required in a testing environment only. It is not necessary when deploying the adapter in a production environment.

How to perform this task

1. Create SCA component
   a. Go to the **Business Integration** perspective of WebSphere Integration Developer. In the **Business Integration** tab, right-click the **JDBCEMD** module, and select **Open With > Assembly Editor**.
   b. The Assembly Diagram window is displayed with the module's Export component in view. To create a new component, click the first icon (Component) in the left-hand (vertical) frame of the window.
   c. Click the first icon in the new menu of icons (it has hover-help that reads **Component with no implementation type**). The cursor changes to the placement icon.
   d. Click the palette to add the new component to the Assembly Diagram window.
2. Create a component with no implementation type
   a. Click and drag the module's Export component to the new component. This draws a wire from the Export component to the new component.
   b. In the Add Wire window, click **OK**.
   c. The new component displays in the Assembly Diagram window with a wire that connects it to the module's Export component.

   Click **File > Save** to save the assembly diagram.
3. Generate an implementation for the new component
   a. Right-click on the **Component with no implementation type** and select **Generate implementation . . . → Java**.
   b. Select **Default package** and click **OK**. This creates an endpoint for the inbound module.
   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 assembly diagram.

   For more information, refer to the WebSphere Integration Developer user guide at http://www.ibm.com/software/integration/wid.

Chapter 8. Deploying the module

To deploy the module to the application server, export the adapter project as an enterprise archive (EAR) file, install the module, and add any configuration properties that were not set in the enterprise service discovery wizard.

Exporting the project as an EAR file

Using the enterprise service discovery wizard, export the adapter project that you have created as an EAR file. By creating an EAR file, you capture all of the contents of your adapter project in a format can be easily deployed to the application server.

Before you begin

Before you can export the project as an EAR file, you must have selected and configured business objects, and generated artifacts.

About this task

Your project file is a J2EE Connector project in your workspace in WebSphere Integration Developer. To export the project to your local file system as an EAR file, perform the following procedure.

How to perform this task

1. In WebSphere Integration Developer, on the Export panel in the Select window, choose **EAR file** from the list and click **Next**.
2. In the EAR Export window, select the business integration module in the **EAR project** field. The module name **JDBCEMD** now has the suffix **App**.
3. Type or browse to the location where you want the EAR file to be created. Click all the check boxes to include everything you created in the EAR file. Click **Finish**.

Result

The adapter project is exported to an EAR file.

What to do next

Use the server administrative console to install the module. This deploys the module to the application server.

Installing the module

Installing the adapter project is the last step of the deployment process. When you install the adapter project on the server and run it, the adapter, which is embedded as part of the project module, runs as part of the installed application.

Before you begin

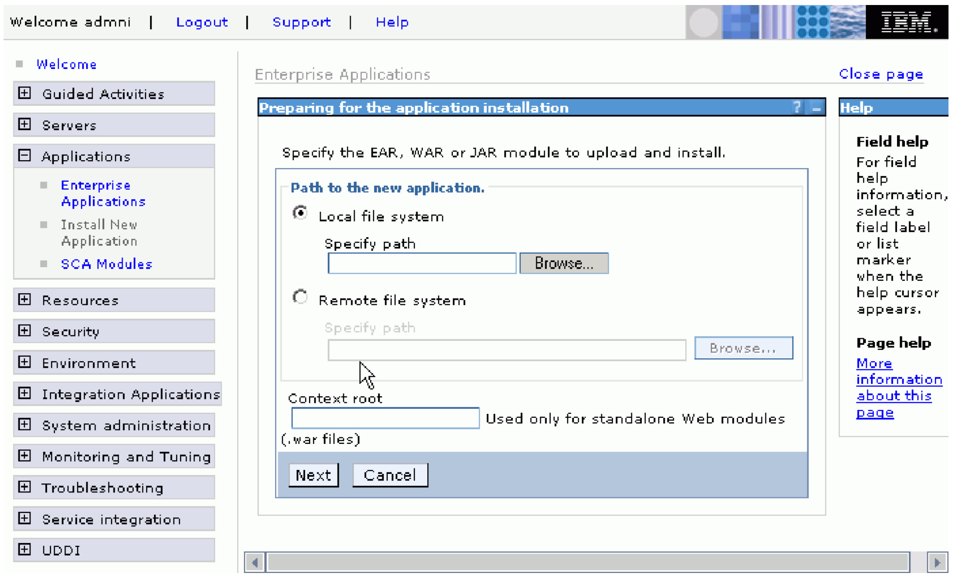
You must have exported your project module as an EAR file before installing the adapter project.

About this task

To install the adapter module, perform the following procedure. For more information on clustering adapter project applications, see http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp.

How to perform this task

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**.

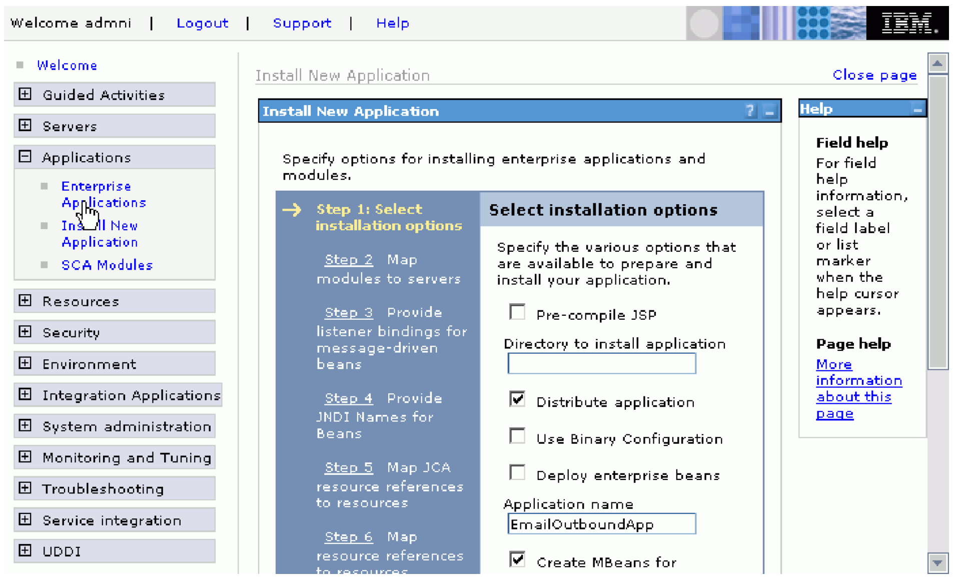


Specify the EAR, WAR or JAR module to upload and install.

Used only for standalone Web modules (.war files)

Preparing for the application installation window

3. Click **Browse** to locate your EAR file and click **Next**.
4. **Optional:** If you are deploying to a clustered environment, click **Next** until you reach Step 2: Mapping modules to servers, then select **Modules** and then the name of the server cluster and click **Apply**. Note: Adapter instances are replicated in a clustered server environment when enableHASupport is set to true. Do not change the value of enableHASupport for single server environments. **Note:** Adapter instances are replicated in a clustered server environment when **enableHASupport** is set to true. Do not change the value of **enableHASupport** for single server environments.
5. Click **Next** until you reach Step 6: Map resource reference to resources.



Install New Application

Specify options for installing enterprise applications and modules.

→ Step 1: Select installation options

Step 2 Map modules to servers

Step 3 Provide listener bindings for message-driven beans

Step 4 Provide JNDI Names for Beans

Step 5 Map JCA resource references to resources

Step 6 Map resource references to resources

Select installation options

Specify the various options that are available to prepare and install your application.

[ ] Pre-compile JSP

Directory to install application

[✓] Distribute application

[ ] Use Binary Configuration

[ ] Deploy enterprise beans

Application name: EmailOutboundApp

[✓] Create MBeans for

Install New Application window

6. Select **SCA Auth Alias** from the select authentication data entry list.
7. Select the check box for the module and click **Apply**.
8. Click **Next**. A summary of all of the installation options is displayed.
9. Verify that all options are correct and click **Finish**.
10. Confirm that the application was installed successfully.
11. Click the **Save to Master Configuration** link at the end of the list of installation messages.
12. Click **Save**.

Result

The project is now deployed and the Enterprise Applications window for the deployed application is displayed.

What to do next

If you want to set or reset resource adapter, managed connection factory, activation specification, or data transformation properties, or you would like to cluster adapter project applications, you should do that using the WebSphere Process Server administrative console before configuring troubleshooting tools.

Setting or changing configuration setting from the administrative console

Once the adapter project is deployed on the server, if you want you can reconfigure properties that the adapter uses to set up a communication channel to a specific database application. Then you can configure troubleshooting tools and start the adapter.

About this task

Normally, the configuration properties are set using the enterprise service discovery wizard when you create your adapter project. You can reset the properties using the administrative console on the server, or you can just note whether the properties are populated according to the values you set in the enterprise service discovery wizard.

Setting resource adapter properties

Resource adapter properties consist of logging and tracing, bidirectional language support, and activities specific to the adapter, such as the default configuration properties of the adapter. You can reset the properties using the administrative console, or just check that the values match what you set using the enterprise service discovery wizard.

How to perform this task

1. Open the adapter project
   a. In the administrative console, in the Configuration pane, the name of your adapter project appears. Under the **Related Items** heading, select **Connector Modules**.
   b. The project RAR file name is displayed. Click the check box next to the file name.
   c. Under **Additional Properties**, select **Resource Adapter**.
2. Edit the Resource adapter property values
   a. Your adapter project is displayed under **General Properties** → **Name**. Under **Additional Properties**, click **Custom properties**.
   b. A list of the Resource adapter properties and their values is displayed. Edit the property values as needed. See the properties and their descriptions in "Resource adapter properties" in the "Reference" section.

      Note: In the **Custom properties** list, at the adapter level, find the **DatabaseVendor** property. This property is required for starting the application. Enter a value for this property if it is blank, and save it.

Setting managed (J2C) connection factory properties

Managed (J2C) connection factory properties are used at run time to create an outbound connection instance with an enterprise information system. You can reset the properties using the administrative console, or just check that the values match what you set using the enterprise service discovery wizard.

How to perform this task

1. Open the adapter project.
   a. In the administrative console, in the Configuration pane, the name of your adapter project is displayed. Under the **Related Items** heading, select **Connector Modules**.
   b. The project RAR file name is displayed. Click the check box next to the file name.
   c. Under **Additional Properties**, select **Resource Adapter**.
2. Edit Managed (J2C) connection factory property values.
   a. Your adapter project is displayed under **General Properties > Name**. Under **Additional Properties**, click **J2C connection factories**.
   b. The adapter project name is displayed along with the JNDI name specified in the EJB project. Click the check box next to your adapter's factory connection.

   c. The J2C connection factory properties and values are shown. Edit the values for the properties as needed. For details about these properties see “Managed (J2C) connection factory properties” in the “Reference” section.

Setting activation specification properties for the EIS

Activation specification properties are required for inbound processing. You can reset the properties using the administrative console, or just check that the values match what you set using the enterprise service discovery wizard.

How to perform this task

1. Open the adapter project
   a. In the administrative console, in the Configuration pane, the name of your adapter project appears. Under the **Related Items** heading, select **Connector Modules**.
   b. The project RAR file name is displayed. Click the check box next to the file name.
   c. Under **Additional Properties**, select **Resource Adapter**.
2. Your adapter project is displayed under **General Properties > Name**. Under **Additional Properties** click **J2C activation specifications**.

   For details about these properties see “Activation specification properties” in the “Reference” section.

Chapter 9. Configuring troubleshooting tools

Configure the troubleshooting tools to suit your requirements. Enable logging for the adapter to control the status of event processing. Enable the Common Event Infrastructure to collect diagnostic information about your adapter. Set tracing levels to determine the level of the information captured in the adapter log and trace files. Install IBM Support Assistant to gain quick access to support-related information along with serviceability tools for problem determination for IBM software products.

Enabling tracing with the Common Event Infrastructure (CEI)

Enable tracing and control the level of detail in the adapter trace by configuring the Common Event Infrastructure (CEI).

Before you begin

Before you enable tracing with CEI, complete the following tasks:
- Enable the diagnostic trace service.
- Publish the IBM WebSphere Adapters event definitions file to the CEI catalog before you can set these event definitions.

For instruction on how to do these tasks, refer to the CEI documentation located on the Web site for your server:
- For WebSphere Process Server: http://www.ibm.com/software/integration/wps
- For WebSphere Enterprise Service Bus: http://www.ibm.com/software/integration/wsesb

To enable tracing and control the level of trace detail, use the following procedure.

How to perform this task
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 General Properties area, click **Change Log Detail Level** and then select **com.ibm.j2ca.*** for the adapter components. There is a subcomponent for each adapter type, as described in the following table.

Adapter	Package Name
WebSphere Adapter for Email	com.ibm.j2ca.email.*
WebSphere Adapter for Flat Files	com.ibm.j2ca.flatfile.*
WebSphere Adapter for FTP	com.ibm.j2ca.ftp.*
WebSphere Adapter for JDBC	com.ibm.j2ca.jdbc.*
WebSphere Adapter for JD Edwards EnterpriseOne	com.ibm.j2ca.jde.*
WebSphere Adapter for SAP Software	com.ibm.j2ca.sap.*
WebSphere Adapter for Siebel Business Applications	com.ibm.j2ca.siebel.*

5. Select the component that matches your adapter. Each adapter component has two subcomponents, one for logging and one for CEI. They are:

- *subcomponent_name*.log.*adapter_ID*
- *subcomponent_name*.cei.*adapter_ID*

  For example, com.ibm.j2ca.siebel.cei.*adapter_ID1*. For each instance of a deployed adapter, the system shows a separate ID.

6. Select the CEI adapter ID that you want to enable.
7. From the list, choose the level of business object detail to capture in service component events:
   - **off**. Turn CEI off.
   - **fine**. Turn CEI on but publish none of the business object payload. This corresponds to the event control detail level of Empty in WebSphere Integration Developer.
   - **finer**. Turn CEI on and publish only the payload description for the business object. This corresponds to the event control detail level of Digest in WebSphere Integration Developer .
   - **finest**. Turn CEI on and publish all of the business object payload. This corresponds to the event control detail level of Full in WebSphere Integration Developer.
   - **all**. Same as **finest**.

   For information on what each event content level means (Empty, Digest and Full), and for more information on using the Common Base Event model and the Common Event Infrastructure, refer to the documentation for your process server.

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.

How to perform this task

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. Select 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 a specific adapter, select its package name.

Adapter	Package Name
WebSphere Adapter for Email	com.ibm.j2ca.email
WebSphere Adapter for Flat Files	com.ibm.j2ca.flatfile
WebSphere Adapter for FTP	com.ibm.j2ca.ftp
WebSphere Adapter for JDBC	com.ibm.j2ca.jdbc
WebSphere Adapter for JD Edwards EnterpriseOne	com.ibm.j2ca.jde
WebSphere Adapter for SAP Software	com.ibm.j2ca.sap
WebSphere Adapter for Siebel Business Applications	com.ibm.j2ca.siebel

7. Click the package name and 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.

Changing the log and trace 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. To keep the adapter log and trace information separate from other processes, use the administrative console to change the file names.

About this task

You can change the log and trace file names at any time after the adapter module has been deployed to an application server.

You can change the log configuration statically or dynamically. Static configuration changes affect applications when you start or restart the application server. Dynamic or run time configuration 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.

How to perform this task

1. In the navigation pane, click **Enterprise Applications**.
2. Click the name of the adapter application. This is the name of the EAR file for the adapter, without the .ear file extension. For example, if the EAR file is named Accounting_OutboundApp.ear, then click **Accounting_OutboundApp**.
3. Click **Connector Modules**.
4. Select the adapter by clicking the name of the RAR file for the adapter. The RAR files are listed in the following table.

Adapter	RAR File Name
WebSphere Adapter for Email	CWYEM_Email.rar
WebSphere Adapter for Flat Files	WYFF_FlatFile.rar
WebSphere Adapter for FTP	CWYFT_FTPFile.rar
WebSphere Adapter for JDBC	CWYBC_JDBC.rar
WebSphere Adapter for JD Edwards EnterpriseOne	CWYED_JDE.rar
WebSphere Adapter for SAP Applications	CWYAP_SAPAdapter.rar CWYAP_SAPAdapterTX.rar

Adapter	RAR File Name
WebSphere Adapter for Siebel Business Applications	CWYEM_Siebel.rar

5. Click the name of the resource adapter.
6. In the Custom Properties area, specify the file names:
   - To change the log file name, type the name in the **Value** field for **logFilename**. By default, this log is in the SystemOut.log file.
   - To change the trace file name, type the name in the **Value** field for **traceFilename**. By default, this log is in the trace.log file.
7. To have static configuration changes take effect, stop and then restart the process server.

Installing or upgrading IBM Support Assistant

IBM Support Assistant (ISA) is a free, local software serviceability workbench that helps you resolve questions and problems with IBM software products. Install plug-ins for the products you have installed. It provides quick access to support-related information along with serviceability tools for problem determination. Installing and upgrading it is simple and straightforward.

About this task

IBM Support Assistant provides the following services:
- Symptom-based data collection
- Access to IBM support information, IBM newsgroups, and other resources through a federated search interface (one search, multiple resources)
- Easy access to IBM educational materials
- Easy access to IBM product home pages, product support pages, and product forums or newsgroups through convenient links
- A tools framework and update manager to easily update and install ISA plug-ins and tools
- Fast resolution of problem management records through electronic submission of critical system data to IBM

You can install and run both version 2 and version 3 of IBM Support Assistant on a single computer, to get support for a broad range of IBM solutions.

To install and upgrade IBM Support Assistant, use the following procedure.

How to perform this task

1. Go to the IBM Support Assistant Web page at:

   http://www.ibm.com/software/support/isa/
2. Follow the directions on the Web page to download ISA version 3.0, and then to extract, install, and use the tool.
3. Start ISA.
4. Open the **Updater** component.
5. On the **Upgrades** tab, upgrade ISA to version 3.0.1 or higher.
6. On the **New Products and Tools** tab, install the plug-ins for your adapter. Select the plug-in for your adapter from the list for the WebSphere brand. There is an optional language pack plug-in for each adapter, which enables you to see adapter-specific information in languages other than English.

Chapter 10. Administering the adapter

Use the administrative console of the server to start, stop, and troubleshoot the adapter.

Starting the adapter

To start an adapter that has a status of Stopped, use the administrative console. By default, an adapter starts automatically when the server starts.

Before you begin

The administrative console of the server must be running in order to complete this task.

To start the adapter, use the following procedure.

How to perform this task

1. On the Enterprise Applications page, click **Applications** → **Enterprise Applications**.
2. Select the check box of the adapter that you want to start.
3. Click **Start**.

Result

The status of the adapter changes to Started and a message stating that the adapter started displays at the top the page.

Use the administrative console of the server to stop the adapter.

Stopping the adapter

Use the administrative console of the server to stop an adapter.

Before you begin

The administrative console of the server must be running in order to complete this task.

To stop the adapter, use the following procedure.

How to perform this task

1. On the Enterprise Applications page, click **Applications** → **Enterprise Applications**.
2. Clear the check box of the adapter you want to stop.
3. Click **Stop**.

Result

The status of the adapter changes to Stopped and a message stating that the adapter stopped displays at the top the page.

Use the administrative console of the server to troubleshoot the adapter.

Troubleshooting and support

Common troubleshooting techniques and self-help information help you identify and solve problems quickly. If necessary, follow the procedures for contacting IBM Software Support.

Exception: 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 self help resources of IBM Software Support to get the most current support information, to obtain technical documentation, to download support tools and fixes, and prevent problems with WebSphere Adapter for JDBC. The self help resources also help you diagnose problems with the adapter and contact IBM Software Support.

The software support Web site for WebSphere Adapters at http://www.ibm.com/software/integration/wbiadapters/supp provides the following resources:

- Flashes (alerts from technical support)
- Technotes

  You can get a list of technotes for WebSphere Adapters at http://www.ibm.com/support/search.wss?rs=695&tc=SSMKUK
- Authorized program analysis reports (APARs)
- Technical information including the product information center, manuals, IBM Redbooks™, and whitepapers.
- Educational offerings
- *IBM Software Support Handbook*

Register at the site to use My Support to create a customized support page for your use.

Contacting IBM Software Support

IBM Software Support provides support for WebSphere Adapters either online or by phone. Gathering information about the problem before you contact IBM Software Support can dramatically increase support responsiveness.

Before you begin

If you think your problem is defect-related, IBM Software Support provides assistance. Before contacting IBM Software Support, your company must have an active IBM software maintenance contract, and you must be authorized to submit problems to IBM. The type of software maintenance contract that you need depends on the type of product you have:

- For IBM distributed software products (including, but not limited to, Tivoli, Lotus®, and Rational® products, as well as DB2 and WebSphere products that run on Windows, Linux®, or UNIX operating systems), you must be enrolled in Passport Advantage®. You can enroll in one of the following ways:

  Online

  Go to the Passport Advantage Web page (http://www-306.ibm.com/software/support/pa.html), and click **How to Enroll**.

  By phone

  For the phone number to call in your country, go to the contacts page of the IBM Software Support Handbook on the Web (http://techsupport.services.ibm.com/guides/contacts.html), and click the name of your geographic region.
- For IBM eServer™ software products (including, but not limited to, DB2 and WebSphere products that run in zSeries®, pSeries®, and iSeries™ environments), you can purchase a software maintenance agreement by working directly with an IBM sales representative or an IBM Business Partner. For more information about support for eServer software products, go to the IBM Technical Support Advantage Web page (http://www-03.ibm.com/servers/eserver/techsupport.html).

If you are not sure what type of software maintenance contract you need, call 1-800-IBMSERV (1-800-426-7378) in the United States or, from other countries, go to the contacts page of the IBM Software Support Handbook on the Web (http://techsupport.services.ibm.com/guides/contacts.html), and click the name of your geographic region for phone numbers of people who provide support for your location.

About this task

The IBM Software Support Handbook contains detailed information about the service and support of your IBM products. Read the handbook at http://techsupport.services.ibm.com/guides/handbook.html.

To contact IBM Software Support, use the following procedure.

How to perform this task

1. Describe your problem and gather background information. When explaining a problem to a support specialist, be as specific as possible. Include all relevant background information so that the specialists can help you solve the problem efficiently. To save time, know the answers to these questions:
   - What software versions were you running when the problem occurred? Include the version of the operating system as well as related products.
   - Has the problem happened before, or is this an isolated problem?
   - What steps led to the failure?
   - Can the problem be recreated? If so, what steps led to the failure?
   - Have any changes been made to the system such as to the hardware, operating system, networking software, and so on?
   - Are you currently using a workaround for this problem? If so, be prepared to explain it when you report the problem.
   - Do you have logs, traces, and messages that are related to the problem symptoms? IBM Software Support is likely to ask for this information.
2. Determine the business impact of your problem. When you report a problem, you will be asked to supply a severity level. Therefore, you need to understand and assess the business impact of the problem you are reporting. Use the criteria described in the following table.

   Table 24. Severity criteria for problem reporting

Severity	Description
1	**Critical business impact**: You are unable to use the program, resulting in a critical impact on operations. This condition requires an immediate solution.
2	**Significant business impact**: The program is usable but is severely limited.
3	**Some business impact**: The program is usable with less significant features (not critical to operations) unavailable.
4	**Minimal business impact**: The problem causes little impact on operations, or a reasonable circumvention to the problem has been implemented.

3. Submit your problem to IBM Software Support. You can submit your problem in the following ways:
   - **Online**. Go to the Submit and track problems page on the IBM Software Support site http://www.ibm.com/software/support/probsub.html Enter your information into the appropriate problem submission tool.

- **By phone**. For the phone number to call in your country, go to the contacts page of the IBM Software Support Handbook on the Web (http://techsupport.services.ibm.com/guides/contacts.html), and click the name of your geographic region.

Result

If the problem you submit is for an unreported software defect or for missing or inaccurate documentation, IBM Software Support creates an Authorized Program Analysis Report (APAR). The APAR describes the problem in detail and tracks its resolution.

What to do next

Whenever possible, IBM Software Support provides a workaround for you to implement until the APAR is resolved and a fix is delivered. IBM publishes resolved APARs on the product support Web pages daily, so that other users who experience the same problem can benefit from the same resolution.

Solutions to some common problems

Some of the problems you may encounter running the Adapter for JDBC with your database are provided, along with solutions and workarounds. These problems and solutions are also documented as technotes on the software support Web site.

Attribute not created in business object for Oracle column of NCHAR type

Problem

When a business object is generated from an Oracle database object, an attribute is not created in the business object for columns of type NCHAR.

Cause

The JDBC driver returns the type of the column as `Other`, which is not supported by enterprise service discovery.

Solution

Use the Business Object Designer tool to manually add attributes for supported types to the business object.

Class loader violation occurs when starting JDBC enterprise service discovery

Problem

It is not possible to use JDBC enterprise service discovery after using a connection to the database in the Data perspective. At the end of the second panel of the enterprise service discovery wizard, the following exception is generated:
com.ibm.adapter.framework.api.ImportException
Reason: class loading constraint violated (class:
oracle/jdbc/driver/OracleConnection
method: getWrapper()Loracle/jdbc/OracleConnection;) at pc:0

The error occurs in both of the following situations:

- When you establish a connection to the database through the enterprise service discovery, an error occurs when you attempt to connect to the database from the Data perspective.
- When you establish a connection to the database through the Data Perspective, the error occurs when you attempt to connect to database through enterprise service discovery.

Cause

The error occurs because the Data Perspective and enterprise service discovery use their own class loaders. Once the DLL, which is the native library used by the JDBC driver, is loaded in the Data Perspective, it cannot be loaded again in enterprise service discovery. JVMs have an inherent restriction that only allows one class loader to load native libraries at any given time. So if class loader A loads DLL B, then no other class loader can load DLL B until class loader A is released and garbage is collected. Because you cannot really control garbage collection, it usually means that if you want to load DLL B with another class loader, you need to restart the JVM. This limitation is a known one and is documented for WebSphere® Application Server.

Solution

The only solution is to restart WebSphere Integration Developer when this error occurs.

Closed connection error occurs when using XA with Oracle 10g

Problem

When the Adapter for JDBC is used to perform an XA transaction using Oracle 10g, the adapter generates a closed connection exception: javax.resource.ResourceException: Closed Connection.

Cause

This is a known issue with the Oracle 10g database driver. The following bug has been filed with Oracle for this issue: 3488761 Connection closed error from OracleConnection.getConnection() - 10G drivers.

Solution

The bug has been fixed in the Oracle 10g Release 2 driver. As a workaround, you can use the Oracle 9i JDBC thin drivers to connect to the database for XA transactions.

Error occurs while starting a transaction on Oracle

Problem

When the Adapter for JDBC is used to perform an XA transaction using Oracle database, the following error is generated: WTRN0078E: An attempt by the transaction manager to call start on a transactional resource has resulted in an error. The error code was XAER_RMERR.

Cause

For the Oracle database server to support XA transactions, some commands need to be run.

Solution

Two scripts that are included in the Oracle directory should be run. This activity will likely need to be performed by your Oracle database administrator, because you must be logged into Oracle as SYSOPER or SYSDBA in order to have the necessary permissions for these scripts to work. The scripts are:

```
<ORACLE_HOME>javavm\install
file: initxa.sql
file: initjvm.sql
```

The initxa.sql script configures the database for XA. Once it runs successfully, your database is configured for XA. The script might run successfully the first time you try. Unfortunately, it probably will not run successfully because some of the database's memory spaces are too small.

To fix this, run the initjvm.sql script. It will probably fail too, but in doing so, it will indicate which parameters need to be adjusted. The parameters are stored in this file:

```
<ORACLE_HOME>\database
file: init<DATABASE_SID>.ora
```

The following table shows two parameters that typically need to be increased. Your particular database configuration might require adjustment of different parameters.

Table 25.

Parameter Name	Minimum Value
java_pool_size	12000000
shared_pool_size	24000000

Using the adapter to connect to DB2 on IBM z/OS with a JDBC (Type 2 or Type 4) universal driver

Problem and Cause

DB2 on z/OS supports querying stored procedure metadata by using positional index by default and not using column name, which the Adapter for JDBC uses. The solution provides the steps for using the WebSphere Adapter for JDBC with DB2 on the z/OS platform.

Solution

To connect to DB2® on z/OS® using the Adapter for JDBC or using DB2 Connect™ with JDBC API, ensure that the following connection requirements are met:

- The physical representation of the universal JDBC driver is the db2jcc.jar file. Ensure that the path to this file is set in the class path.
- Database URL: To determine whether you are using the Type 2 or Type 4 driver, review the form of the connection:

  Type 2: jdbc:db2:database
  (For example: jdbc:db2:MyDB, where MyDB is the database name)

Type 4: jdbc:db2://server:port/database
(For example: jdbc:db2://9.182.15.129:50000/MyDB,, where MyDB is the database name)

- Driver class: com.ibm.db2.jcc.DB2Driver.

  The driver class for both Type 2 and Type 4 drivers is the same.
- Set the path of the db2jcc_license_cisuz.jar file in the class path.

  The license JAR file is the same for both Type 2 and Type 4 drivers. Access to DB2 for z/OS and DB2 for i5/OS® servers requires a valid DB2 Connect™ license. DB2 clients do not provide direct connectivity to zSeries and iSeries servers without a DB2 Connect license.

  For more information on DB2 Connect licensing and usage, refer to the following pages:
  http://www-128.ibm.com/developerworks/db2/library/techarticle/0303zikopoulos1/0303zikopoulos1.html
  http://www-128.ibm.com/developerworks/db2/library/techarticle/0301zikopoulos/0301zikopoulos.html

There might be issues with importing metadata for stored procedures using enterprise service discovery. To use stored procedures and import metadata from DB2 using the Adapter for JDBC, DB2 needs to be reconfigured as described in the following steps. Follow these steps in addition to the steps provided above:

- Apply the following APARs on DB2: PQ62695, PQ55393, PQ56616, PQ54605, PQ46183, and PQ62139.
- If you want to use stored procedures with the adapter, follow the steps below, which are part of the fix for PQ62695. This fix introduces stored procedures that provide the ability to generate a result set that corresponds to the Schema Metadata APIs documented in the JDBC and ODBC specification.

  These procedures will be used by the JDBC and ODBC drivers provided in the DB2 Universal Driver. Follow these steps to enable support for stored procedures:

  1. Apply the APAR.
  2. Check the value of the DESCSTAT variable in the ZPARM assembly job DSNTIJUZ. If the value of the DESCSTAT variable is NO, change it to YES.

     Note: The default for DESCSTAT is NO on V7 but was changed to YES on V8.
  3. Reassemble and re-initialize the ZPARM module.
  4. Run the JCL job named DSNTIJMS. You can find this member in the db2prefix.SDSNSAMP data set.
  5. Restart DB2.

Using transactions in lieu of wrapper business objects

This applies only to the English language version.

Problem

The WebSphere Business Integration Adapter for JDBC did not expose support for transactions to the integration broker (WebSphere InterChange Server). This created a problem when users wanted to ensure that two unrelated objects, for example, a customer and an order, were created in their enterprise information system (EIS) at the same time.

To do this, two separate requests had to be sent, and if the adapter went down between requests, only one of the objects was created. Because of limitations of the WebSphere InterChange Server adapter interface, the solution was to allow users to send multiple requests in a single batch to the adapter through a wrapper object. This wrapper object was a dummy container to hold 1..n child business objects, and a commit to the backend database was issued only after each of the child business objects was processed successfully.

Solution

The new WebSphere Adapter for JDBC (JCA) supports local and global transactions. Now, when you want to send a request that simultaneously creates a customer and an order, for example, you can start a transaction from an adapter client (for example, a mediation or business process) and send the requests as you see fit. Once you have sent everything you want to send, you commit the transaction. There is no need for a wrapper business object.

Using XA transactions for outbound support with a remote DB2 database

This provides the steps, database versions, and configuration requirements for XA transaction support using the Adapter for JDBC with a remote DB2 database.

Using XA Transactions on a remote DB2 database

Adding a remote DB2 database

1. Run the db2admin (<DB2_Installpath>\SQLLIB\BIN) command on the DB2 server machine.
2. Open the DB2 Configuration Assistant.
3. Go to **View** → **Advanced View**.

Perform the following four tasks–adding the remote system, adding an instance node, adding a database, and testing the database connection–in the order in which they are described.

Adding the remote system

1. Select **Systems** tab.
2. From the menu, choose **Selected** → **Add System** .
3. In the **System name** field, specify the physical machine, server system, or workstation where the target database is located. The system name on the server system is defined by the DB2SYSTEM DAS configuration parameter. This is the value that you should use.
4. In the **Host name** field, type the host name or Internet Protocol (IP) address where the target database resides.
5. In the **Node name** field, specify a local nickname for the remote node where the database is located. The node name you choose must not already exist in the node directory or the admin node directory.
6. Select the operating system and click **OK**.

Adding an instance node

1. Select the **Instance Nodes** tab.
2. From the menu, choose **Selected** → **Add Instance Node** .

3. In the **System name** field, specify the physical machine, server system, or workstation where the target database is located. Select the system added in the Adding a remote system task.
4. In the **Instance name** field, type the name of the instance (DB2, and so on) where the target database is located.
5. In the **Instance node name** field, specify a unique nickname for the cataloged system (node) where the database is located. The node name you choose must not already exist in the node directory or the admin node directory.
6. Select the operating system and type the host name. Use the same host name as in step 4 of the task: Adding the remote system.
7. Enter the port number on which the remote DB2 instance is running.
8. Click **OK**.

Adding a database

1. Select the **Databases** tab.
2. From the menu, choose **Selected** → **Add Database**.
3. In the **Instance node** field, choose the instance created in the task: Adding an instance node. Specify the name of the database that you are adding in the **Database name** field.
4. In the **Alias** field, specify a local nickname that can be used by applications running on your workstation. If nothing is entered, the alias will be the same as the database name. The alias name should be unique.

   Note: This alias name value should be entered in the XADatabaseName property for the adapter.

Testing the database connection

1. Select the **Databases** tab.
2. Choose the database added in the task: Adding a database.
3. From the menu, choose **Selected** → **Test Connection**.
4. Select the **CLI** check box, type the User Id and Password and click **Test Connection**. This should return a successful connection.

Using XA Transactions with the Adapter for JDBC using Universal Driver

The following versions of software and configuration properties are needed to use XA Transactions with the Adapter for JDBC (JCA) and a remote DB2 database.

- DB2 version: 8.2
- JDBC Driver: UDB driver (db2jcc.jar) Type 4
- XA datasource name: com.ibm.db2.jcc.DB2XADataSource
- XA Database name: This is the remote database alias configured on the local DB2 client.
- Database URL: jdbc:db2://hostname:port/databasename
- JDBC driver class: com.ibm.db2.jcc.DB2Driver

WebSphere Adapters have no separate logs for inbound and outbound operations

Problem

If you create and bind an import and an export to the same adapter instance (with the same AdapterID property) and you name the logs for the two bindings differently, for example: ″a.log″ for the import and ″b.log″ for the export, you will find after deploying the project to WebSphere Process Server, that the resource adapter has only one log, the one named ″b.log.″

Cause

WebSphere® Adapters do not distinguish messaging from inbound and outbound operations, and therefore create only one adapter log file.

Solution

Configure the inbound and outbound operations with the same log name if Import and Export are binding to the same adapter instance. If the inbound and outbound belong to different adapters and each has a different AdapterID, this issue will not exist.

Chapter 11. Quick start tutorial

To gain practical knowledge in setting up and deploying the adapter, complete the tutorial. Everything you need to complete the tutorial is contained in the tutorial. If you have performed the prerequisite tasks (such as installing the adapter), you can complete the tutorial in under an hour.

Introduction

The tutorial provides a complete set of instructions for configuring the adapter so that it can be used by a J2EE component to send requests to the database. The tutorial demonstrates creating a record, which has a parent-child relationship, in a database table. The scenario also shows how to use a stored procedure attached to a business object.

Learning objectives

After completing the tutorial, you should be able to perform the following tasks:

- Create an adapter project in WebSphere Integration Developer
- Discover services and associated business objects from your database and make them part of the adapter project
- Create a deployable module that you install in the WebSphere Process Server test environment
- Test the module to ensure that it operates correctly and to see the results of running the module

Time required

This tutorial should take less than one hour to finish.

Audience

The tutorial is intended for the integration developer who will be configuring the Adapter for JDBC for deployment on WebSphere Process Server or WebSphere Enterprise Service Bus.

Prerequisites

Before you begin the tutorial, make sure you have performed the following tasks:

- Install all of the prerequisite software
- Install the Adapter for JDBC
- Make sure you have all the information (such as user ID and password) to access the database
- Create tables and stored procedures using the following instructions.

Create tables and stored procedures

The following tables and stored procedures must be created in the database to run the tutorial.

- Script for creating the Customer and Address tables:

```
CREATE TABLE CUSTOMER (
          "PKEY" VARCHAR(10) NOT NULL PRIMARY KEY,
          "FNAME" VARCHAR(20),
          "LNAME VARCHAR(20),
          "CCODE" VARCHAR(10);

CREATE TABLE ADDRESS (
          "ADDRID" VARCHAR(10) NOT NULL PRIMARY KEY,
          "CUSTID" VARCHAR(10),
          "CITY" VARCHAR(20),
          "ZIPCODE" VARCHAR(10);
```

- Creating the stored procedure:

  You can create the stored procedure using the IBM DB2 Development Center. In the development center, create a new Java stored procedure using the wizard. Create a stored procedure that does an insert on the Address table.

Tutorial: Create a record in database tables

In this scenario, you create a record using parent and child business objects, and associate a stored procedure with the child business object. First, you create an adapter project, then use the enterprise service discovery wizard to generate Customer and Address business objects. You create a module containing the adapter and the newly generated business objects, and deploy the module to the test environment.

Creating the authentication alias

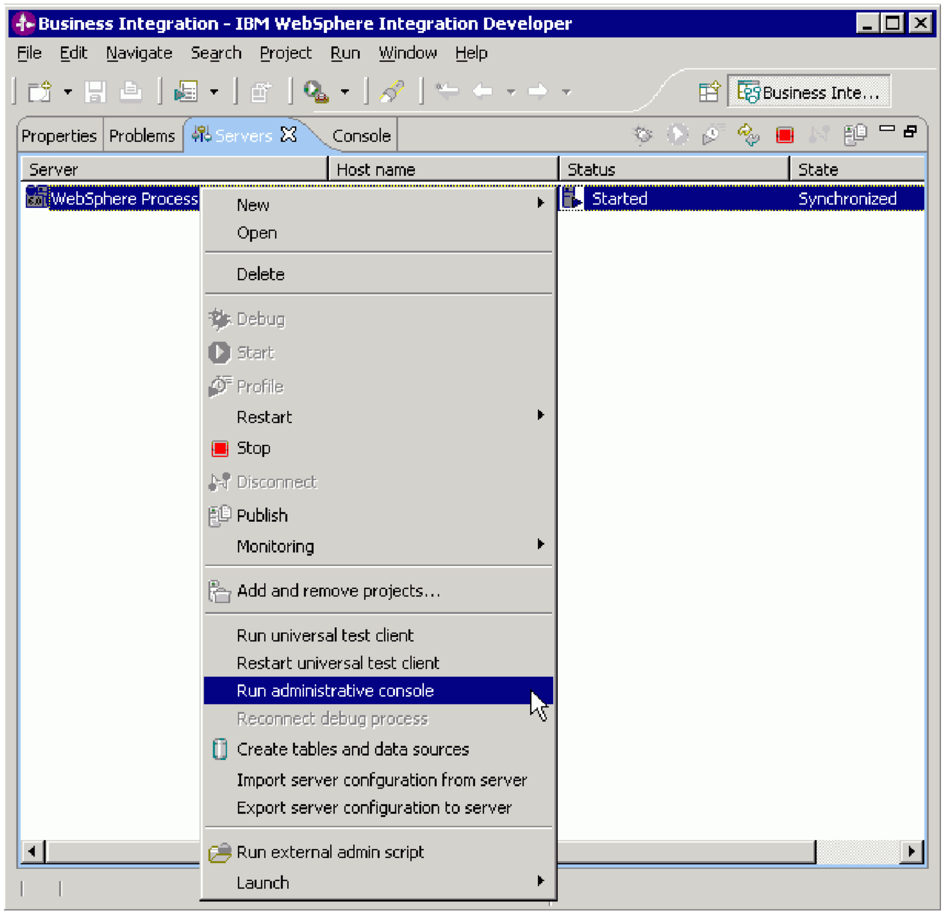
Create the authentication alias on the server by using the server's administrative console. From the administrative console, configure the authentication alias with the user ID and password that enable you to access the database for processing outbound requests.

About this task

Before you configure the adapter, you must create an authentication alias on the server. To create the authentication alias, use the following procedure.

How to perform this task

1. Start WebSphere Integration Developer by clicking **Start** → **Programs** → **IBM WebSphere** → **Integration Developer V6.0.2** → **WebSphere Integration Developer V6.0.2**.
2. If you are prompted to specify a workspace, accept the default value. The workspace is a directory where WebSphere Integration Developer stores your project.
3. When the WebSphere Integration Developer window is displayed, close the Welcome page.
4. Switch to the Business Integration Perspective by clicking **Window** → **Open Perspective**. Then click **Business Integration (default)** and click **OK**.
5. Display the administrative console.
   a. Click the **Servers** tab.
   b. If **WebSphere Process Server v6.0.2** does not show a status of **Started**, right-click **WebSphere Process Server v6.0.2** and click **Start**.
   c. Right-click **WebSphere Process Server v6.0.2** and click **Run administrative console**.



Business Integration - IBM WebSphere Integration Developer
File Edit Navigate Search Project Run Window Help
Business Inte...
Properties Problems Servers Console
Server Host name Status State
WebSphere Process Started Synchronized
New
Open
Delete
Debug
Start
Profile
Restart
Stop
Disconnect
Publish
Monitoring
Add and remove projects...
Run universal test client
Restart universal test client
Run administrative console
Reconnect debug process
Create tables and data sources
Import server confguration from server
Export server configuration to server
Run external admin script
Launch

Figure 24. Run the administrative console on the server

d. Log in to the administrative console by typing **admin** and clicking **Login**.

6. In the administrative console, click **Security** → **Global security**.

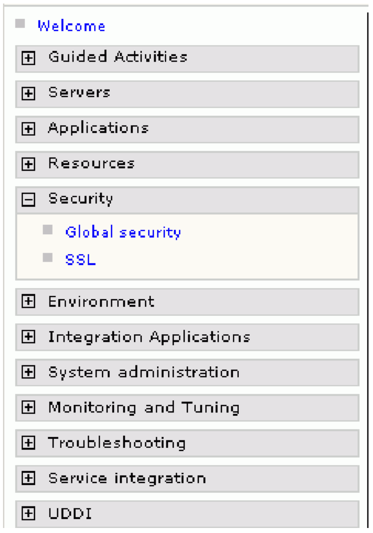
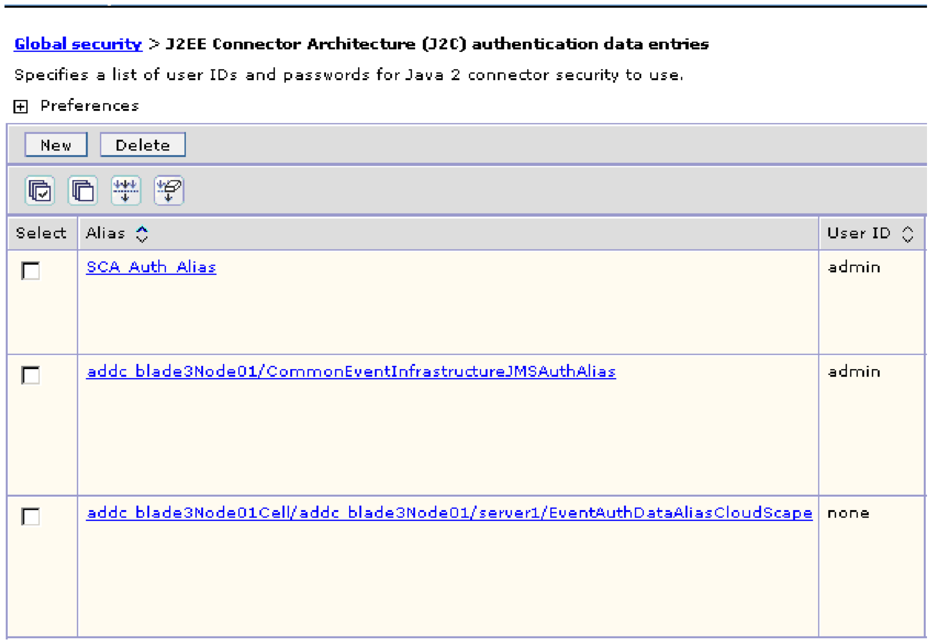


Figure 25. Selecting Global security

7. Under the Authentication heading, click **JAAS Configuration** → **J2C Authentication data**.

8. Select the authentication alias **SCA_Auth_Alias**.


Global security > J2EE Connector Architecture (J2C) authentication data entries

Specifies a list of user IDs and passwords for Java 2 connector security to use.

Select	Alias	User ID
	SCA_Auth_Alias	admin
	addc_blade3Node01/CommonEventInfrastructureJMSAuthAlias	admin
	addc_blade3Node01Cell/addc_blade3Node01/server1/EventAuthDataAliasCloudScape	none

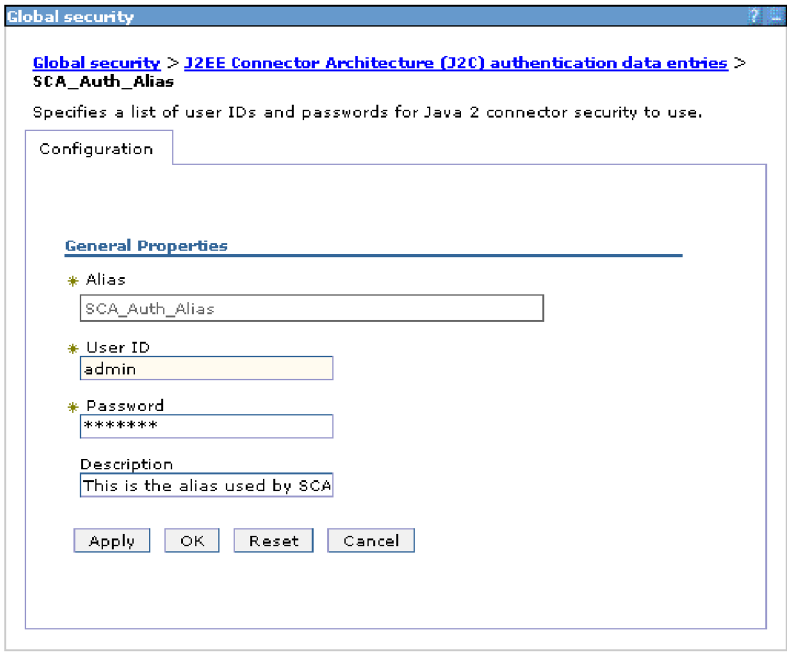
Figure 26. Selecting the authentication alias

- If the **SCA_Auth_Alias** entry does not already exist, click **New** and create it by typing it in the **Alias** field of the General properties window.
- Otherwise, click **SCA_Auth_Alias**.

The authentication alias name is case-sensitive.

9. Type the user id and password that are required to connect to the database.

Type the password in the format that your database requires, such as all uppercase.


Global security

Global security > J2EE Connector Architecture (J2C) authentication data entries > SCA_Auth_Alias

Specifies a list of user IDs and passwords for Java 2 connector security to use.

Configuration

General Properties

* Alias: SCA_Auth_Alias

* User ID: admin

* Password: *******

Description: This is the alias used by SCA

Figure 27. Entering authentication data

10. Click **OK**.

    Make note of the name as it appears in the Alias list. You will use this name in subsequent configuration windows.
11. Click **Save** and then click **Save** again. The changes will take affect when you restart the server.

Result

You have created an authentication alias that you will use when you configure the adapter properties under "Generating Artifacts."

Creating the adapter project in WebSphere Integration Developer

To begin the process of creating a module to communicate with a database, you create an adapter project. The adapter project contains the adapter itself plus other related artifacts. You create the project by importing the resource adapter archive (RAR) file, which was copied to your local file system during installation, into the Connector Projects folder in WebSphere Integration Developer.

1. In WebSphere Integration Developer, switch to the J2EE perspective:
   a. Click **Window** → **Open Perspective** → **Other**.
   b. Click **J2EE**.

      If **J2EE** is not displayed, select the **Show all** check box, click **J2EE**, and click **OK**.

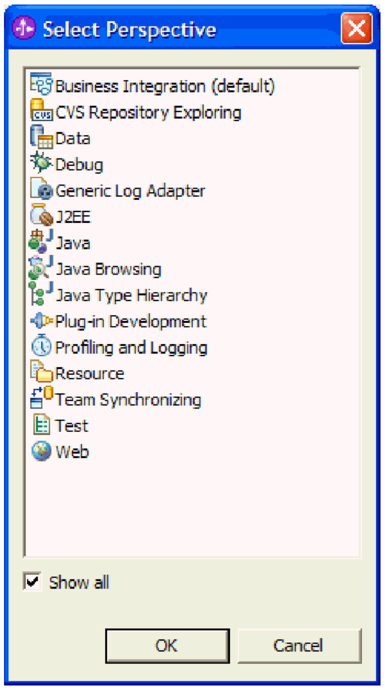


Figure 28. Selecting J2EE from the Select Perspective list

c. If you see the Confirm Enablement window, select **Always enable capabilities and don't ask me again**.

d. Click **OK**.

2. Import the RAR file by right-clicking **Connector Projects** and clicking **Import** → **RAR file**.

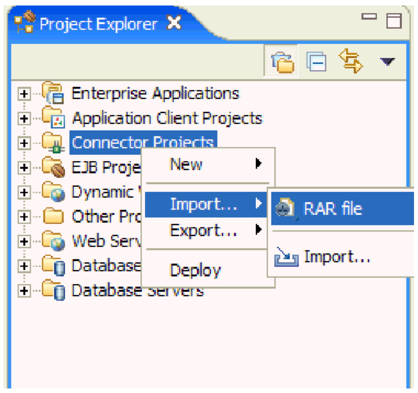
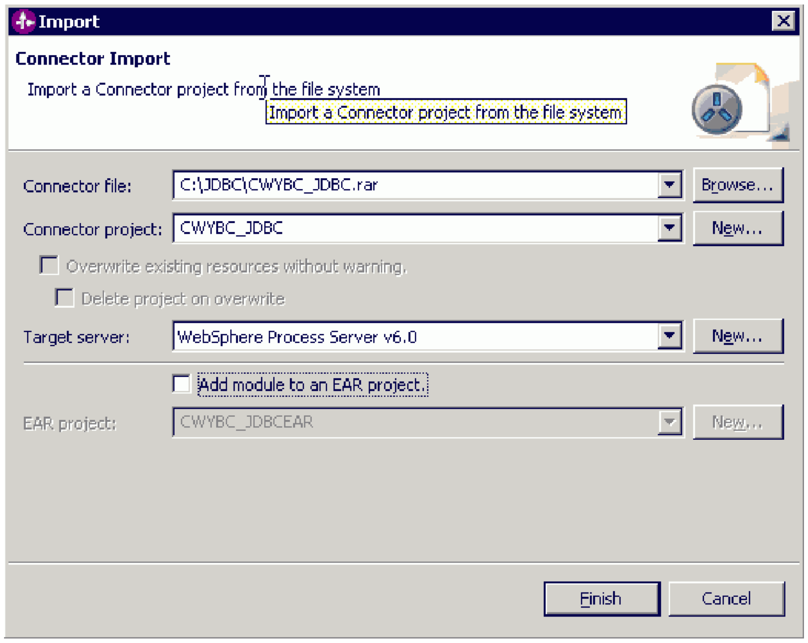


Figure 29. Importing the RAR file

3. Find the RAR file on your local file system by clicking **Browse** and navigating to the directory in which the Adapter for JDBC was installed.


Import

Connector Import

Import a Connector project from the file system

Import a Connector project from the file system

Connector file: C:\JDBC\CWYBC_JDBC.rar

Connector project: CWYBC_JDBC

Overwrite existing resources without warning.

Delete project on overwrite

Target server: WebSphere Process Server v6.0

Add module to an EAR project.

EAR project: CWYBC_JDBCEAR

Figure 30. Selecting the RAR file from the installation directory

4. Accept the default setting (**CWYBC_JDBC**) for **Connector project**.

   The adapter project has the same name as the RAR file.

   If a project named CWYBC_JDBC.rar already exists in this workspace, the name in the **Connector project** field has a number appended to it (for example, CWYBC_JDBC1).
5. Accept the default value in the **Target server** field.

   The default value is the test environment for WebSphere Process Server, which is installed as part of WebSphere Integration Developer.
6. Clear the **Add module to an EAR project** check box.

   Notice that the EAR project field becomes unavailable after you remove the check mark.
7. Click **Finish**.

Result

A new J2EE adapter project, named CWYBC_JDBC, is created. To see its contents, expand **CWYBC_JDBC**.

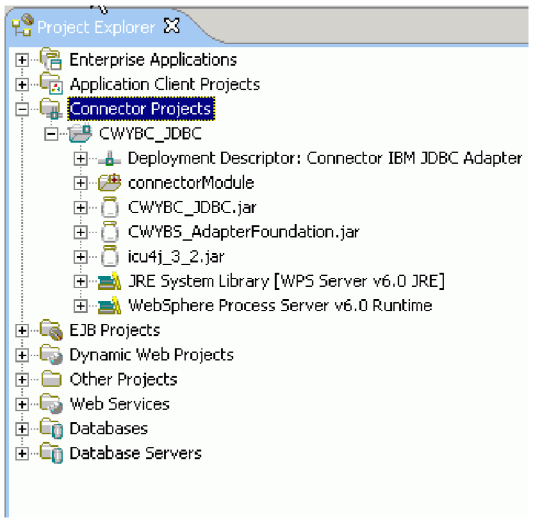


Figure 31. The CWYBC_JDBC adapter project in the Project Explorer window

Adding external dependencies

After you create the project in WebSphere Integration Developer, you need to add the database driver to your adapter project. To accomplish this, add the JDBC driver to the Java build path. Use the external JAR file that is provided with your database.

1. In the J2EE perspective of WebSphere Integration Developer, expand **Connector Projects**. Right-click the adapter project **CWYBC_JDBC** and select **Properties**.

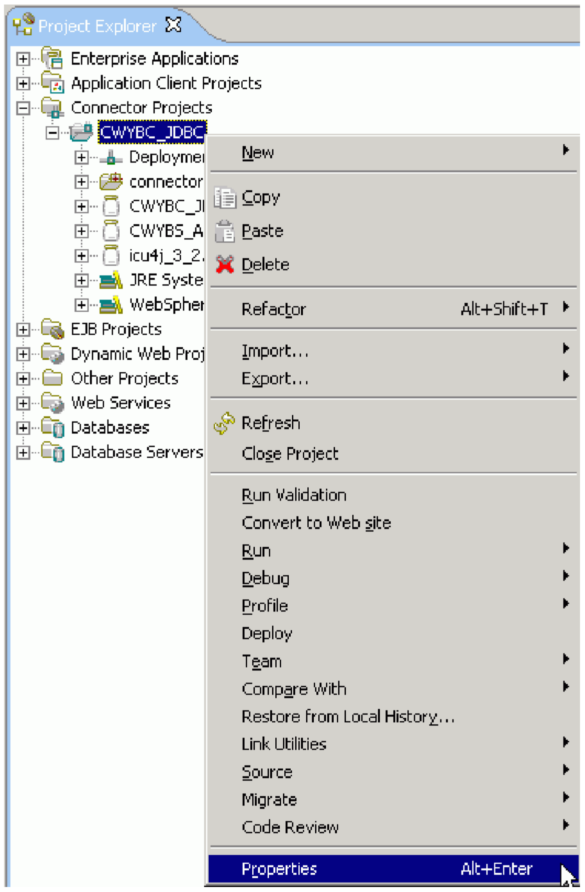


Figure 32. Selecting Properties option for the project

2. In the navigation sidebar, click **Java Build Path**. Select the **Libraries** tab and click **Add External JARs**.

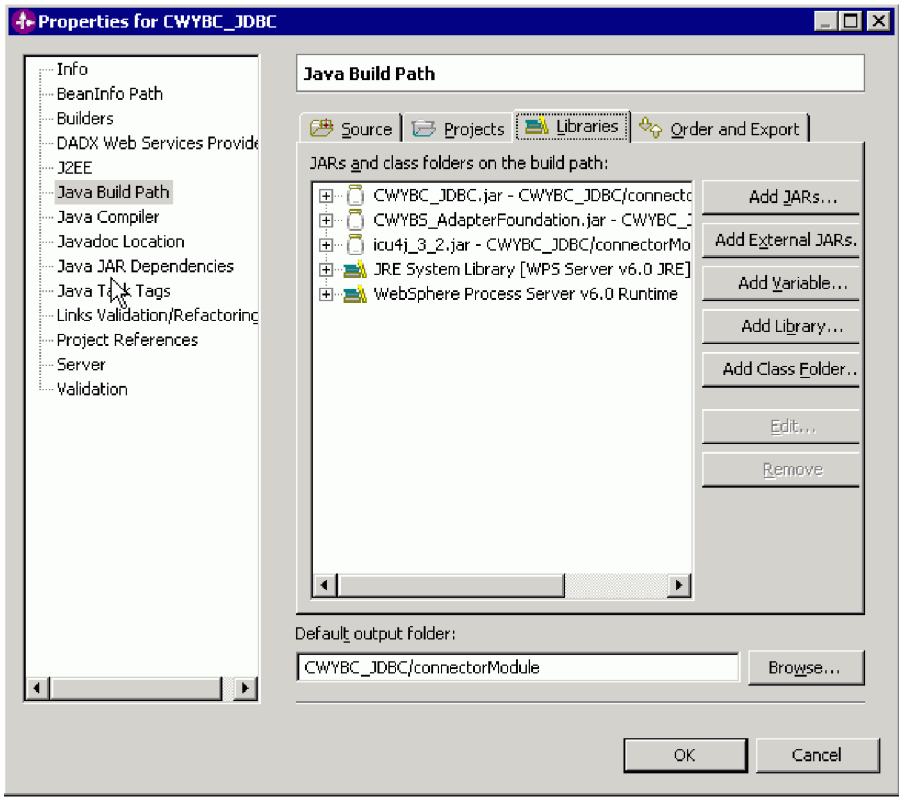

Properties for CWYBC_JDBC
Info
BeanInfo Path
Builders
DADX Web Services Provide
J2EE
Java Build Path
Java Compiler
Javadoc Location
Java JAR Dependencies
Java Task Tags
Links Validation/Refactoring
Project References
Server
Validation
Java Build Path
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
CWYBC_JDBC.jar - CWYBC_JDBC/connectc
CWYBS_AdapterFoundation.jar - CWYBC_J
icu4j_3_2.jar - CWYBC_JDBC/connectorMo
JRE System Library [WPS Server v6.0 JRE]
WebSphere Process Server v6.0 Runtime
Add JARs...
Add External JARs.
Add Variable...
Add Library...
Add Class Folder..
Edit...
Remove
Default output folder:
CWYBC_JDBC/connectorModule
Browse...
OK
Cancel

Figure 33. Adding external JAR file

3. In the JAR Selection window, navigate to the directory on your local file system that contains the JDBC driver JAR file. Click **Open**.

   The JDBC driver JAR file is added to the Java build path.
4. Click **OK**.

Result

Your adapter project contains a reference to the JDBC driver, which you can view in the Project Explorer window of WebSphere Integration Developer. The driver has been added so that the adapter can connect to the database, and business objects can be generated for tables, stored procedures, and so on.

Configuring the adapter for outbound processing

To configure WebSphere Adapter for JDBC for outbound processing, use the enterprise service discovery wizard in WebSphere Integration Developer. Configuration includes setting the connection properties for enterprise service discovery, selecting business objects in the enterprise information system, and generating the business object definitions for outbound processing.

Setting connection properties for enterprise service discovery

You need to start the enterprise service discovery wizard and set the values of the connection properties for your database instance. The enterprise service discovery process requires these properties to connect to the database for discovery and for creating the service description.

1. Start enterprise service discovery.

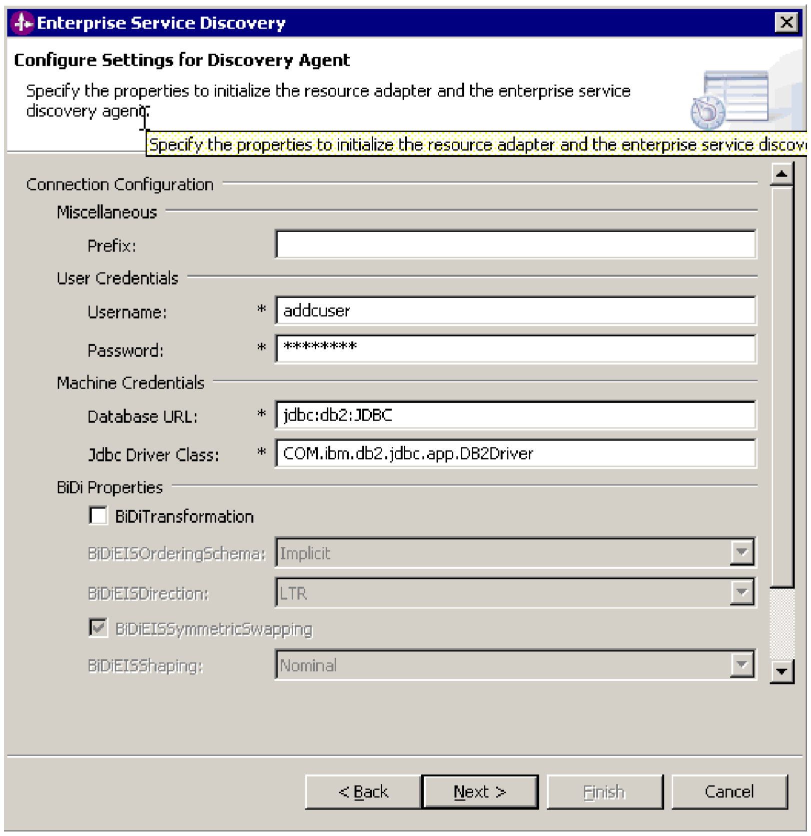
a. In WebSphere Integration Developer, switch to the **Business Integration** perspective by selecting **Window** → **Open Perspective** → **Other.** Click **Business Integration (default)** and click **OK**.

b. Select **File** and right-click **New** → **Enterprise Service Discovery**.

   If **Enterprise Service Discovery** is not displayed, click **New** → **Other**, expand **Business Integration**, and click **Enterprise Service Discovery**. Then click **Next**.

2. In the Enterprise Service Discovery window, select **JDBC EMD Adapter (version 6.0.2) from the 'CWYBC_JDBC' Connector Project** and click **Next**.
3. In the Configure Settings for Discovery Agent window, set the **Connection Configuration** properties required to connect to the database.

   Type the **Username**, **Password**, **Database URL**, and **JDBC Driver Class** and click **Next**. Check your driver documentation for the appropriate values for Database URL and JDBC Driver Class.


Enterprise Service Discovery

Configure Settings for Discovery Agent

Specify the properties to initialize the resource adapter and the enterprise service discovery agent.

Connection Configuration

Miscellaneous

Prefix:

User Credentials

Username: * addcuser

Password: * ********

Machine Credentials

Database URL: * jdbc:db2:JDBC

Jdbc Driver Class: * COM.ibm.db2.jdbc.app.DB2Driver

BiDi Properties

☐ BiDiTransformation

BiDiEISOrderingSchema: Implicit

BiDiEISDirection: LTR

☑ BiDiEISSymmetricSwapping

BiDiEISShaping: Nominal

Figure 34. Configure Settings for Discovery Agent window

4. Set the logging level so that you can see any errors that might arise during configuration.

   The default location for the log file, called JDBCMetadataDiscovery.log, is the .metadata folder in the workspace.

   a. At the bottom of the Configure Settings for Discovery Agent window, click **Show Advanced**. The button changes to **Hide Advanced**.

   b. For **Logging Level**, select **FINEST**.

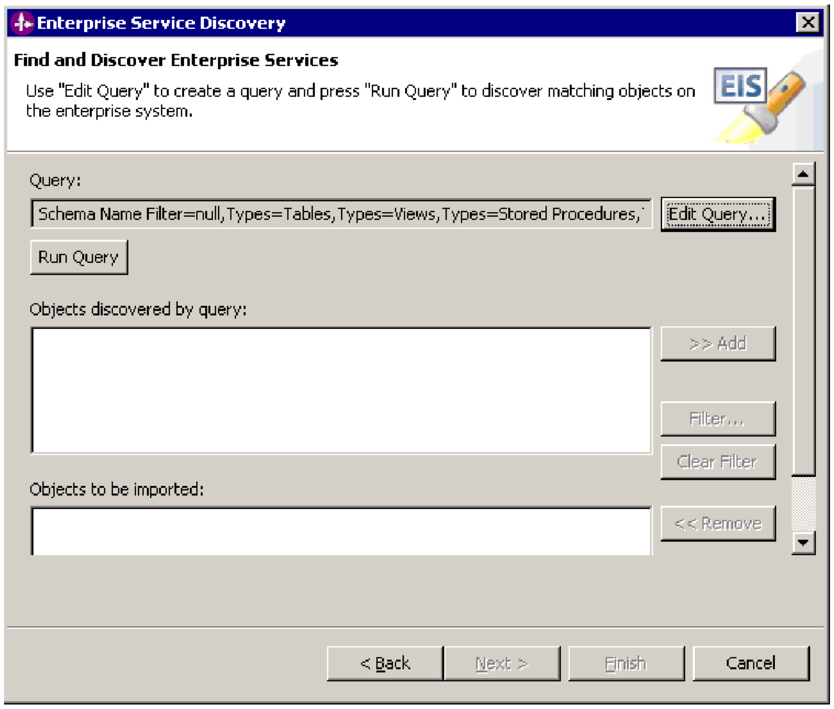
   c. Click **Next**.

Selecting business objects and services

You will run a query to find the database objects, Address and Customer. You will associate a stored procedure with the Address business object. Finally, you will select these objects to import.

1. In the Find and Discovery Enterprise Services window, click **Edit Query**.

   You can use Edit Query to add business object application-specific information. You could use Edit Query to narrow the list of schemas, nodes, or objects displayed in the tree structure. For this scenario, you do not need to use filtering.



Enterprise Service Discovery

Find and Discover Enterprise Services

Use "Edit Query" to create a query and press "Run Query" to discover matching objects on the enterprise system.

Query:

Schema Name Filter=null,Types=Tables,Types=Views,Types=Stored Procedures,

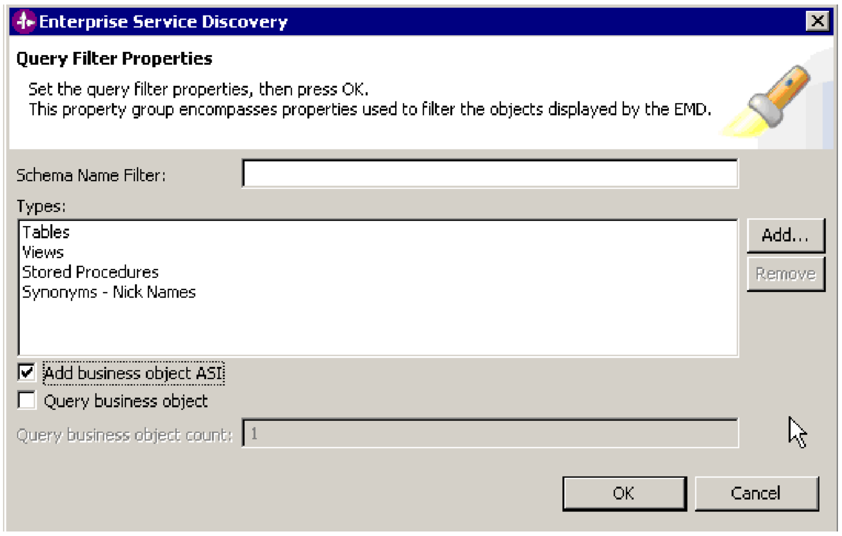
Objects discovered by query:

Objects to be imported:

Figure 35. Find and Discover Enterprise Services window

2. In the Query Filter Properties window, check the **Add business object ASI** check box and click **OK**.

   Then whenever you add an object when you run the metadata query, a window is displayed for entering application-specific information (ASI).


Enterprise Service Discovery

Query Filter Properties

Set the query filter properties, then press OK.
This property group encompasses properties used to filter the objects displayed by the EMD.

Schema Name Filter:

Types:

Tables
Views
Stored Procedures
Synonyms - Nick Names

[✓] Add business object ASI

[ ] Query business object

Query business object count: 1

Figure 36. Query Filter Properties window

3. In the Find and Discover Enterprise Services window, run the metadata query by clicking **Run Query**.

   The objects discovered by the query are displayed in the top pane. The tables, views, stored procedures, and synonyms/nicknames are sorted by schema name.
4. Search for objects in the list by expanding the **ADDCUSER** schema name and expanding the **Tables** type.
5. Click the **Address** and **Customer** tables, and then click **>>Add** to select the objects for import.

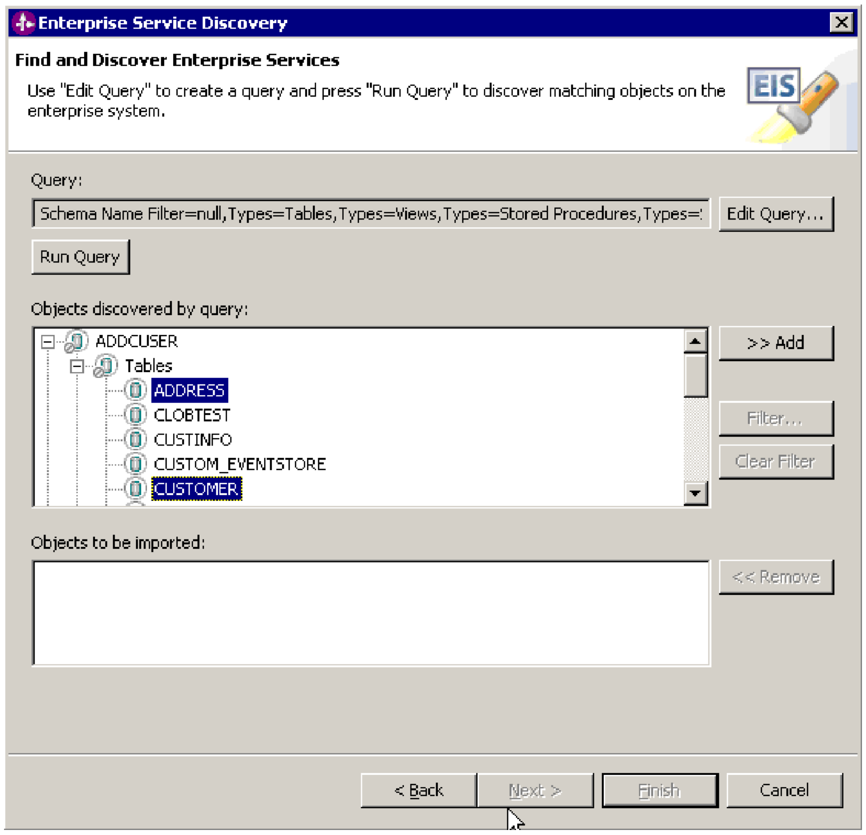

Enterprise Service Discovery
Find and Discover Enterprise Services
Use "Edit Query" to create a query and press "Run Query" to discover matching objects on the enterprise system.
EIS
Query:
Schema Name Filter=null,Types=Tables,Types=Views,Types=Stored Procedures,Types=
Edit Query...
Run Query
Objects discovered by query:
ADDCUSER
Tables
ADDRESS
CLOBTEST
CUSTINFO
CUSTOM_EVENTSTORE
CUSTOMER
>> Add
Filter...
Clear Filter
Objects to be imported:
<< Remove
< Back
Next >
Finish
Cancel

Figure 37. Selecting the objects discovered by the query

The next window is displayed because you chose to add business object ASI in step 2.

6. In the Configuration Parameters for ADDRESS window, click **Add.**

   The **Status Column Name** is used only for logical deletes, and **Status Value** specifies what values to update in the column corresponding to the attribute chosen for Status Column Name. Leave these two parameters as is, because logical deletes are not being performed in this scenario.

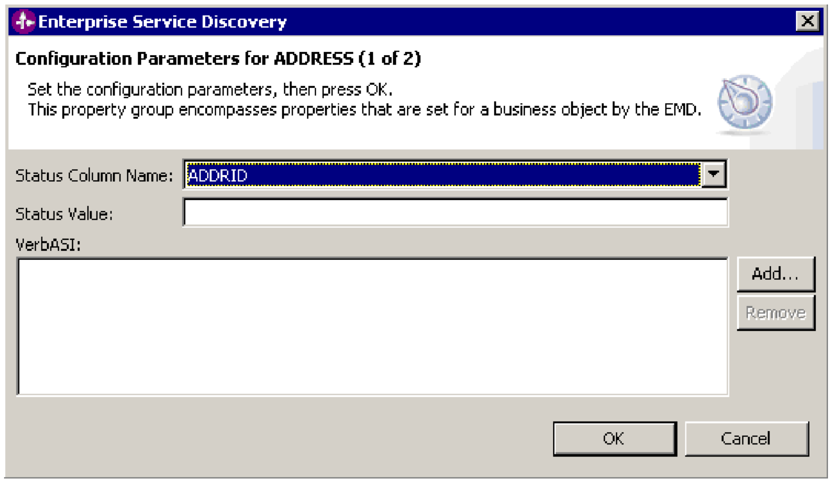

Enterprise Service Discovery
Configuration Parameters for ADDRESS (1 of 2)
Set the configuration parameters, then press OK.
This property group encompasses properties that are set for a business object by the EMD.
Status Column Name: ADDRID
Status Value:
VerbASI:
Add...
Remove
OK
Cancel

Figure 38. Configuration Parameters for ADDRESS window

A list of stored procedures is displayed.

7. Select **CreateSP** and click **OK**. This associates the CreateSP stored procedure application-specific information with the ADDRESS table. When a Create operation is called on the business object, the stored procedure specified in CreateSP is called, and performs the insert operation.

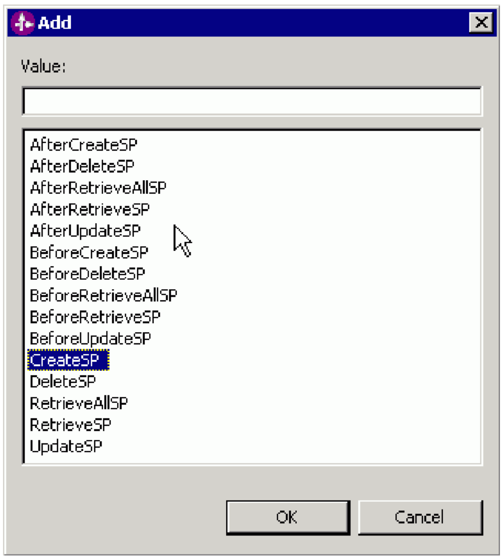
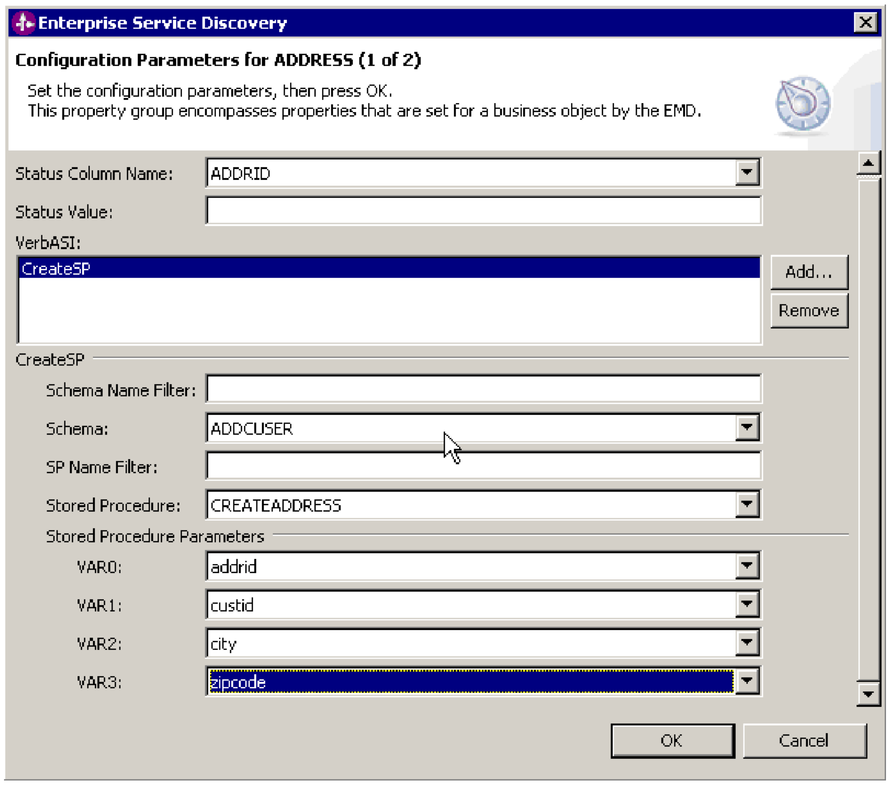


Figure 39. List of stored procedures to add to the Verb ASI

8. In the Configuration Parameters for ADDRESS window, under **CreateSP**, type in values for the **Schema** and **Stored Procedure** fields.

   Use the following values:
   - Schema: ADDCUSER
   - Stored Procedure: CREATEADDRESS

9. When you enter the **Stored Procedure** name, the **Stored Procedure Parameters** fields are displayed. Click **OK.**


Enterprise Service Discovery

Configuration Parameters for ADDRESS (1 of 2)

Set the configuration parameters, then press OK.
This property group encompasses properties that are set for a business object by the EMD.

Status Column Name: ADDRID
Status Value:
VerbASI:
CreateSP

CreateSP
Schema Name Filter:
Schema: ADDCUSER
SP Name Filter:
Stored Procedure: CREATEADDRESS
Stored Procedure Parameters
VAR0: addrid
VAR1: custid
VAR2: city
VAR3: zipcode

Figure 40. Values for CreateSP

The Configuration Parameters for CUSTOMER window is displayed.

When you select more than one business object to add, the window to add business object ASI for the first object is displayed. After you entered the application-specific information, the same window for the second business object is displayed.

10. No stored procedure will be associated with the Customer business object, so no verb application-specific information needs to be added. Click **OK**.
11. The Find and Discover Enterprise Services window shows **ADDRESS** and **CUSTOMER** in the **Objects to be imported** pane. Click **Next**.

Result

The selected objects have been imported for use with the adapter.

Configuring the selected objects

After selecting database objects, you need to specify values for the Metadata selection properties for the import file.

How to perform this task

1. In the Configure Objects window, select **Outbound** for the **Service Type**.

   The **Operations** field lists the objects that the adapter supports for the outbound service type.
2. Do not change the default values for the **Name Space** and **Max Records** fields. Leave **BO Location** blank.

   Name Space is prepended to the business object name to keep the business object schemas logically separated. The default is the Name Space for the metadata schema JDBCASI.xsd.

Max Records represents the maximum number of records to retrieve for a RetrieveAll operation.

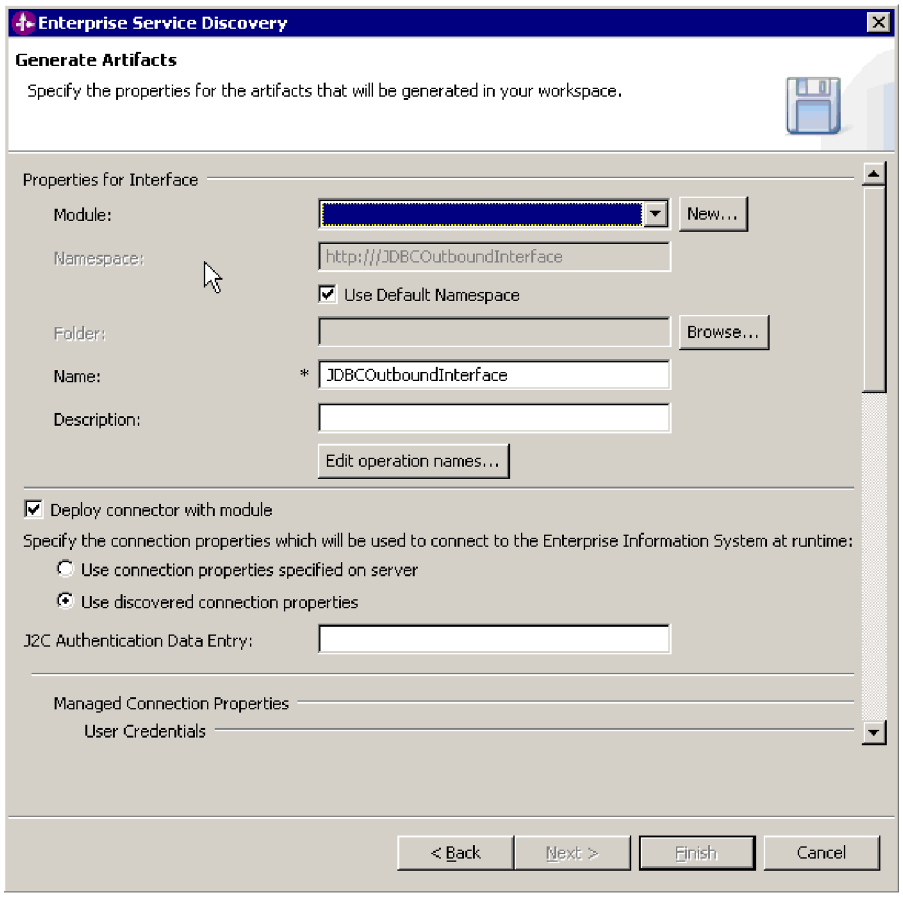
BO Location represents the path to the location where the generated .xsd files are to be stored.

3. Click **Next**.

Generating artifacts

You configure properties that the adapter uses to set up a communication channel to a specific database. You also need to create a new business integration module where all the artifacts and property values can be saved.

1. In the Generate Artifacts window, create a new module by clicking **New**. If the New Integration Project window appears, select **Create a module project** and click **Next**.


Enterprise Service Discovery

Generate Artifacts

Specify the properties for the artifacts that will be generated in your workspace.

Properties for Interface

Module:

Namespace: http:///JDBCOutboundInterface

[✓] Use Default Namespace

Folder:

Name: * JDBCOutboundInterface

Description:

[✓] Deploy connector with module

Specify the connection properties which will be used to connect to the Enterprise Information System at runtime:

○ Use connection properties specified on server

● Use discovered connection properties

J2C Authentication Data Entry:

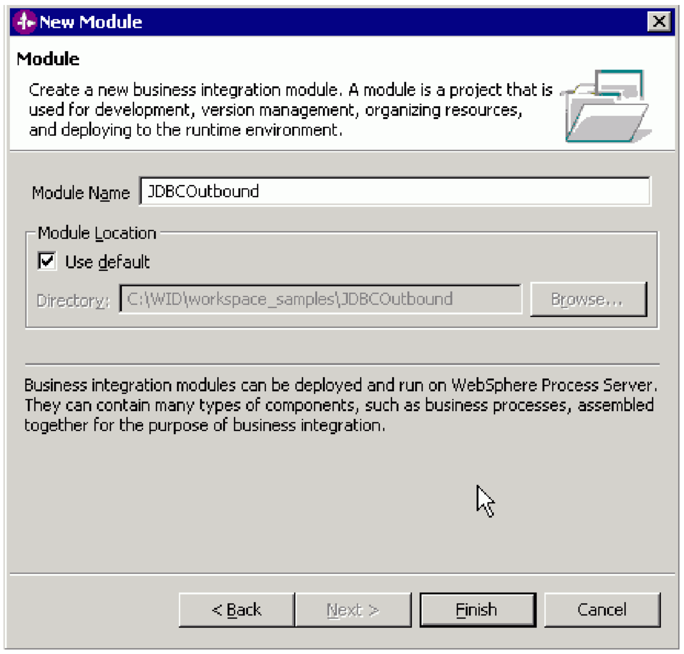
Managed Connection Properties

User Credentials

Figure 41. Generate Artifacts window

2. In the New Module window, type **JDBCOutbound** in the **Module Name** field and click **Finish**.

   Do not change the default **Module Location**.



New Module

Module

Create a new business integration module. A module is a project that is used for development, version management, organizing resources, and deploying to the runtime environment.

Module Name: JDBCOutbound

Module Location

[✓] Use default

Directory: C:\WID\workspace_samples\JDBCOutbound

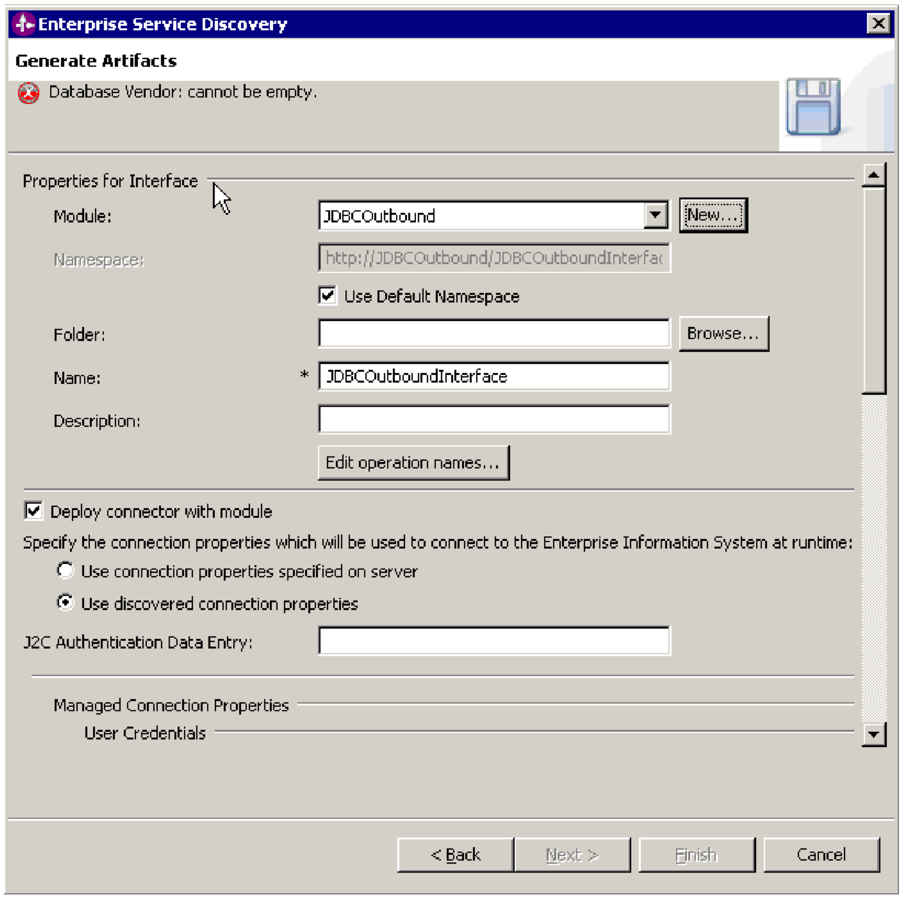
Business integration modules can be deployed and run on WebSphere Process Server. They can contain many types of components, such as business processes, assembled together for the purpose of business integration.

Figure 42. New Module window

3. In the Generate Artifacts window, select **Use discovered connection properties**.

   This enables you to set configuration property values that will be used to connect to the enterprise information system at run time. For outbound processing, the properties are displayed for the Managed (J2C) connection factory and Resource adapter properties. Required property fields are marked with an asterisk.

   For more details on the Adapter configuration properties, see the “Reference” section.


Enterprise Service Discovery

Generate Artifacts

Database Vendor: cannot be empty.

Properties for Interface

Module: JDBCOutbound

Namespace: http://JDBCOutbound/JDBCOutboundInterfac

[✓] Use Default Namespace

Folder:

Name: * JDBCOutboundInterface

Description:

[✓] Deploy connector with module

Specify the connection properties which will be used to connect to the Enterprise Information System at runtime:

( ) Use connection properties specified on server

(●) Use discovered connection properties

J2C Authentication Data Entry:

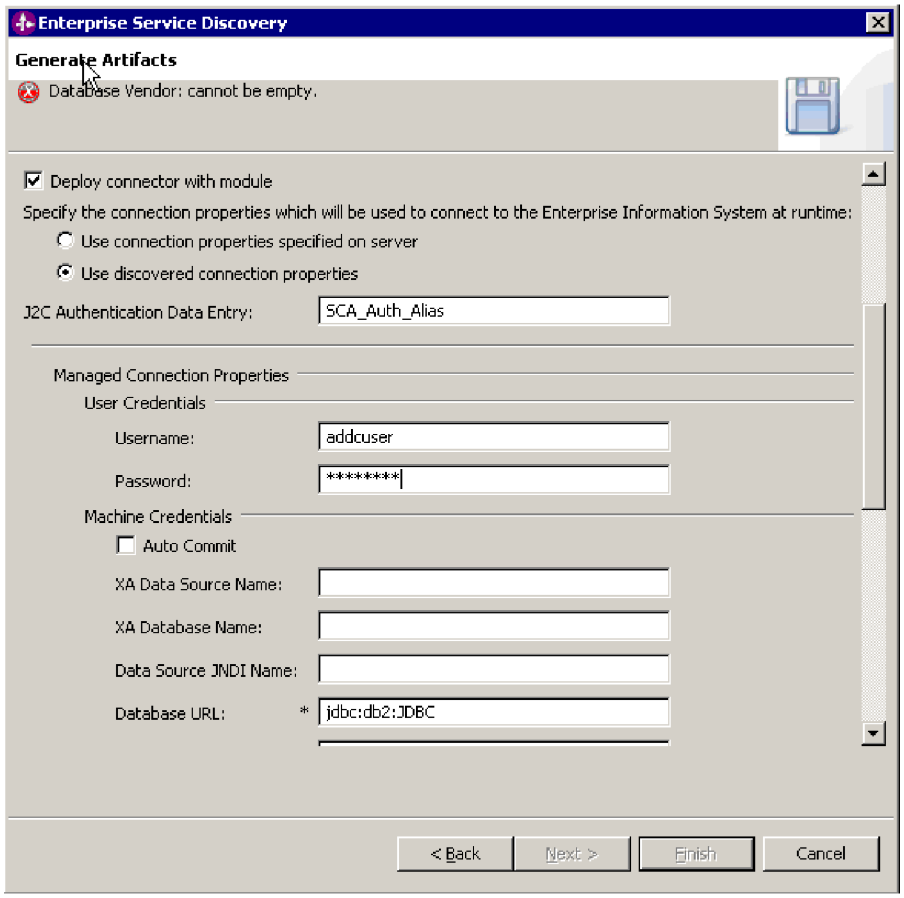
Managed Connection Properties

User Credentials

Figure 43. Use discovered connection properties

4. For the **JDBC Authentication Data Entry**, type **SCA_Auth_Alias**, which is the authentication alias you created earlier.

   The name is case-sensitive.
5. Type the connection properties **Username**, **Password**, **Database URL** and **JDBC Driver Class** that you set after starting the enterprise service discovery wizard.



Enterprise Service Discovery

Generate Artifacts

Database Vendor: cannot be empty.

☑ Deploy connector with module

Specify the connection properties which will be used to connect to the Enterprise Information System at runtime:

○ Use connection properties specified on server

● Use discovered connection properties

J2C Authentication Data Entry: SCA_Auth_Alias

Managed Connection Properties

User Credentials

Username: addcuser

Password: ********

Machine Credentials

☐ Auto Commit

XA Data Source Name:

XA Database Name:

Data Source JNDI Name:

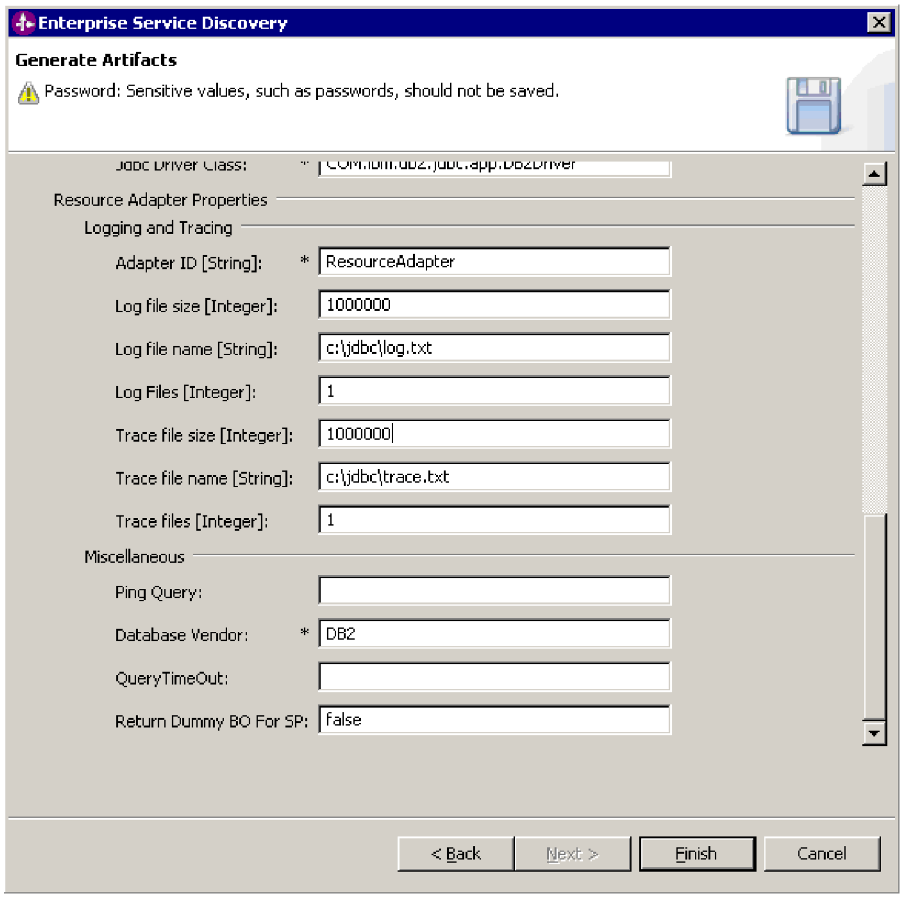
Database URL: * jdbc:db2:JDBC

Figure 44. Set the Alias and Managed connection properties

6. Provide values only for each property field marked with an asterisk.

   You can also set logging and tracing properties if you want the adapter log and trace files to be generated. You can use the property values displayed in the following figure.

7. When you are finished setting required properties, click **Finish**.



Generate Artifacts

Password: Sensitive values, such as passwords, should not be saved.

Jdbc Driver Class:	*	COM.ibm.db2.jdbc.app.DB2Driver

Resource Adapter Properties

Logging and Tracing

Adapter ID [String]:	*	ResourceAdapter
Log file size [Integer]:		1000000
Log file name [String]:		c:\jdbc\log.txt
Log Files [Integer]:		1
Trace file size [Integer]:		1000000
Trace file name [String]:		c:\jdbc\trace.txt
Trace files [Integer]:		1

Miscellaneous

Ping Query:		
Database Vendor:	*	DB2
QueryTimeOut:		
Return Dummy BO For SP:		false

Figure 45. Set Resource adapter properties

Verify the results by comparing your output to the tutorial sample file. In the Business Integration perspective, at the Business Integration pane, expand the **JDBCOutbound** module. Expand the **Data Types** folder to view a list of all the business objects.

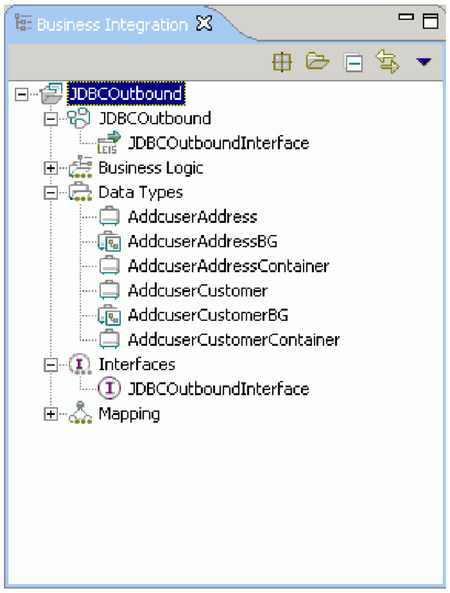


Figure 46. Verify the results of generating artifacts

Linking business objects in a parent-child relationship

After generating artifacts, you need to link the Customer and Address business objects in a parent-child relationship. You will provide additional application-specific information to support the linking of the objects.

How to perform this task

1. Under **Data Types**, double-click **AddcuserCustomer** to open the business object in business object designer.
2. Click the icon for **Add an attribute to a business object**.

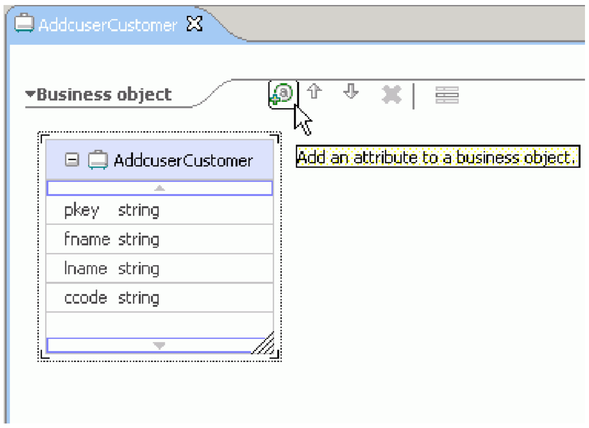


Figure 47. Add an attribute to the business object

This creates a new attribute called **attribute1**, whose type is the AddcuserAddress object.

3. Rename this new attribute by typing **addrObj**.
4. Change the type of the attribute by clicking on the **string** type. A list of types that can be associated with this attribute is displayed. Select the **AddcuserAddress** type.

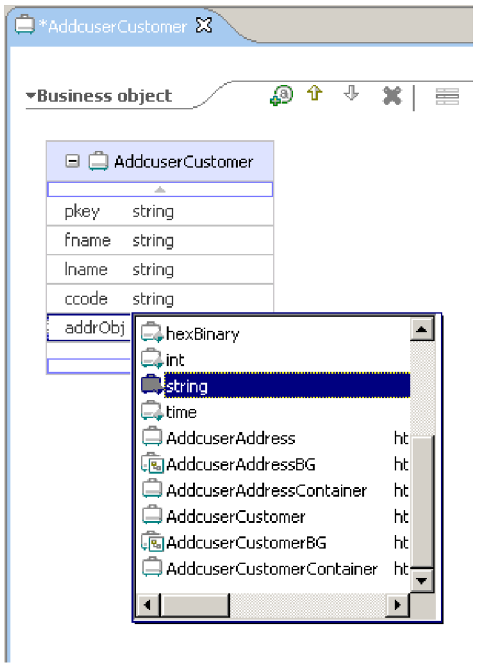


Figure 48. Change type of attribute

5. Indicate this Address child object is a multiple cardinality type; that is, one Customer object can have multiple Address objects.
   a. Select the new attribute **addrObj AddcuserAddress**.
   b. In the **Properties** tab, click **Description.**
   c. Select the **Array** checkbox.

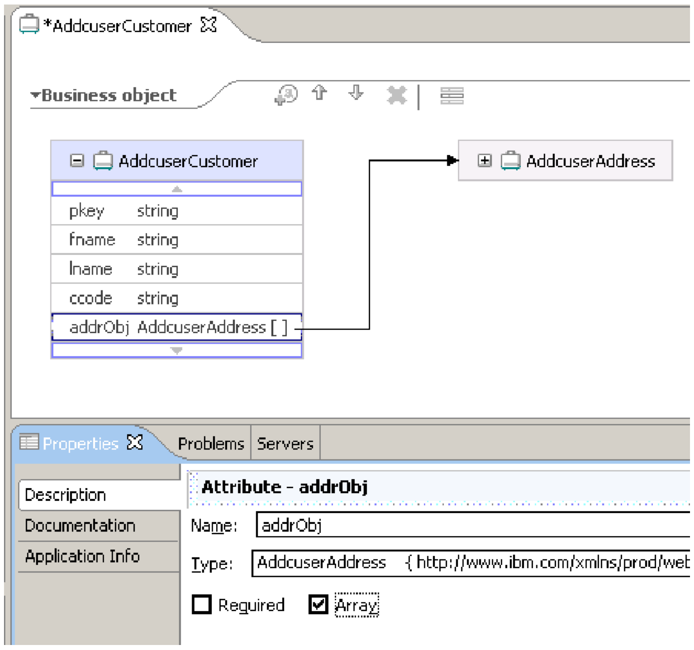


Figure 49. Check Array for multiple cardinality

6. Add the attribute application-specific information for the addrObj AddcuserAddress attribute.

a. In the **Properties** tab, click **Application Info**.

b. Select the **JDBCAttributeTypeMetadata** check box to add attribute application-specific information to the newly added attribute.

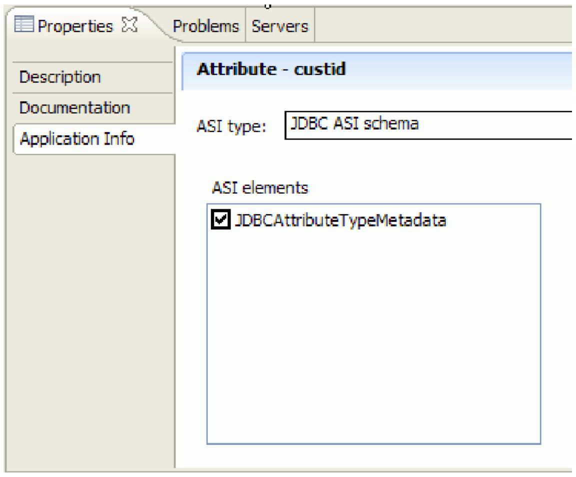


Figure 50. Add attribute application-specific information

c. Under **ASI element properties**, right-click **jdbcasi: JDBCAttributeTypeMetadata**. Select **Add Child** → **jdbcasi:Ownership**.

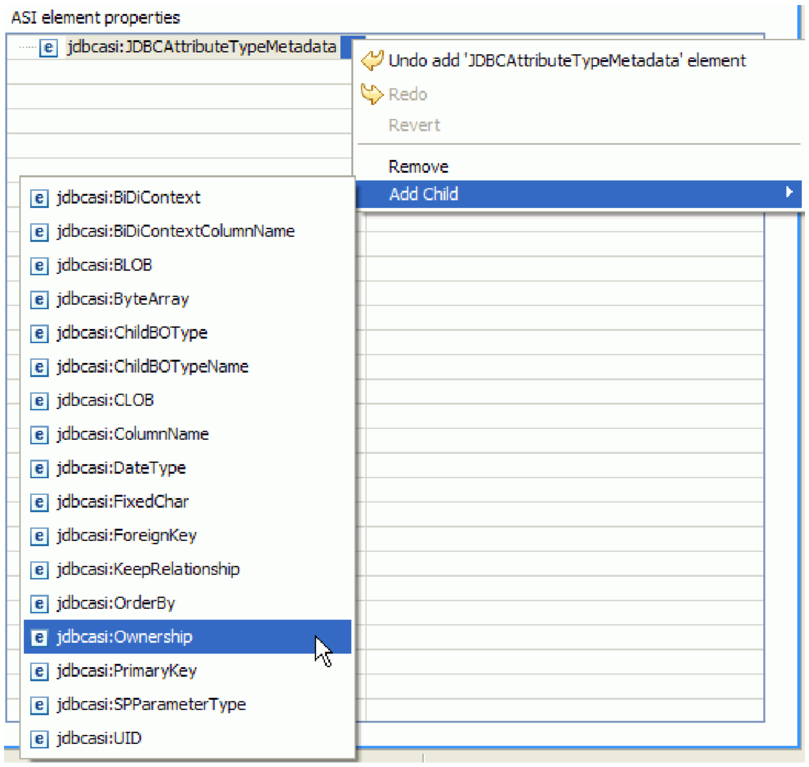
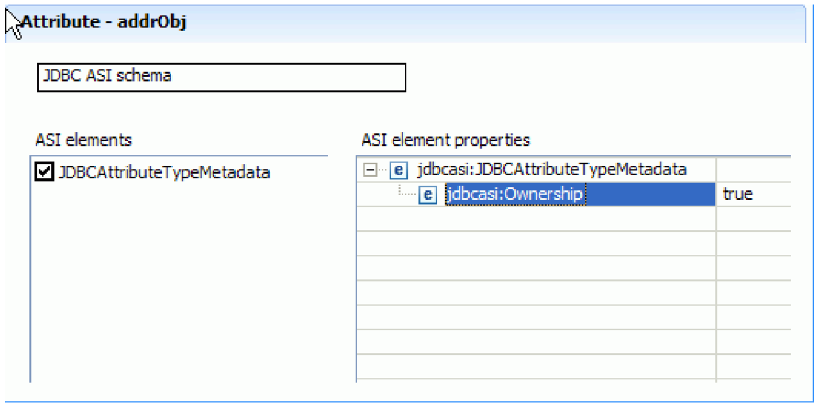


Figure 51. ASI element properties

d. Use the default value of **true** for **jdbcasi:Ownership**. By setting the ownership value to true, you indicate that the parent owns the child; that is, the child cannot exist without the parent.

e. Click **Save** to save the Customer business object.



Attribute - addrObj

JDBC ASI schema

ASI elements

- [✓] JDBCAttributeTypeMetadata

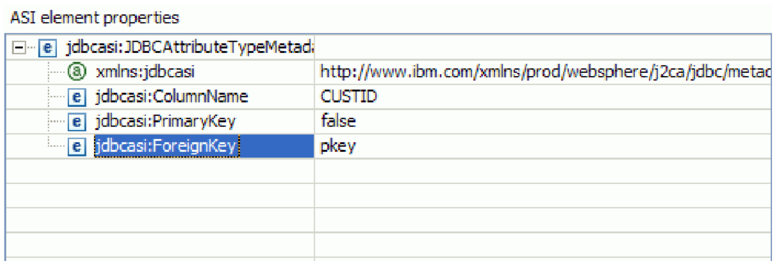
ASI element properties

jdbcasi:JDBCAttributeTypeMetadata	
jdbcasi:Ownership	true

Figure 52. Set ownership

7. Set the foreign key on the AddcuserAddress business object.
   a. Under **Data Types**, double-click **AddcuserAddress** to open this business object in business object designer.
   b. Click the **custid** attribute.
   c. In the **Properties** tab, click **Application Info**.
   d. Right-click **jdbcasi:JDBCAttributeTypeMetadata**. Select **Add Child** → **jdbcasi: ForeignKey**. Type the value **pkey**.

      The pkey value represents the attribute in the Customer object that refers to the primary key in the Customer table.



ASI element properties

jdbcasi:JDBCAttributeTypeMetad	
xmlns:jdbcasi	http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metad
jdbcasi:ColumnName	CUSTID
jdbcasi:PrimaryKey	false
jdbcasi:ForeignKey	pkey

Figure 53. Set foreign key

Deploying the module for testing

The enterprise service discovery process results in a Service Component Architecture (SCA) module that contains an enterprise information system import file. You use WebSphere Integration Developer to access the server and deploy the SCA module to the test environment of the server.

1. In WebSphere Integration Developer, switch to the Servers View by clicking **Windows** → **Show View** → **Servers**.

   Note: Before you start the server, ensure that the database driver JAR file exists in the runtime's lib folder of WebSphere Process Server.
2. Click the **Servers** tab, right-click **WebSphere Process Server v6.0.2** and select **Start**.
3. When the server status is **Started**, right-click the server and select **Add and remove projects**.

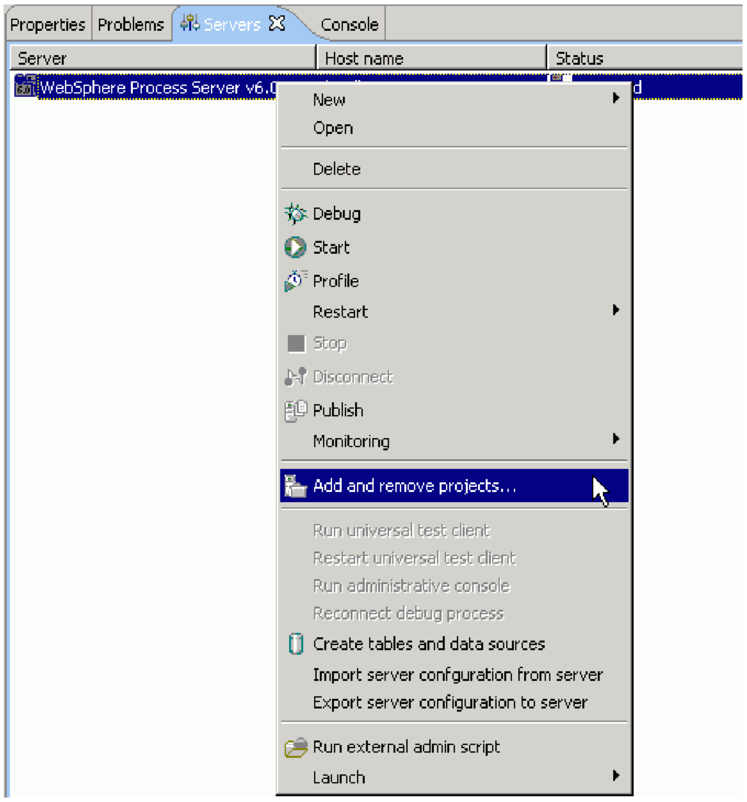

Properties
Problems
Servers
Console
Server
Host name
Status
WebSphere Process Server v6.0
New
Open
Delete
Debug
Start
Profile
Restart
Stop
Disconnect
Publish
Monitoring
Add and remove projects...
Run universal test client
Restart universal test client
Run administrative console
Reconnect debug process
Create tables and data sources
Import server configuration from server
Export server configuration to server
Run external admin script
Launch

Figure 54. Add and remove projects option

4. Select **JDBCOutboundApp** and click **Add**.

   The module moves from the **Available projects** list to the **Configured projects** list.

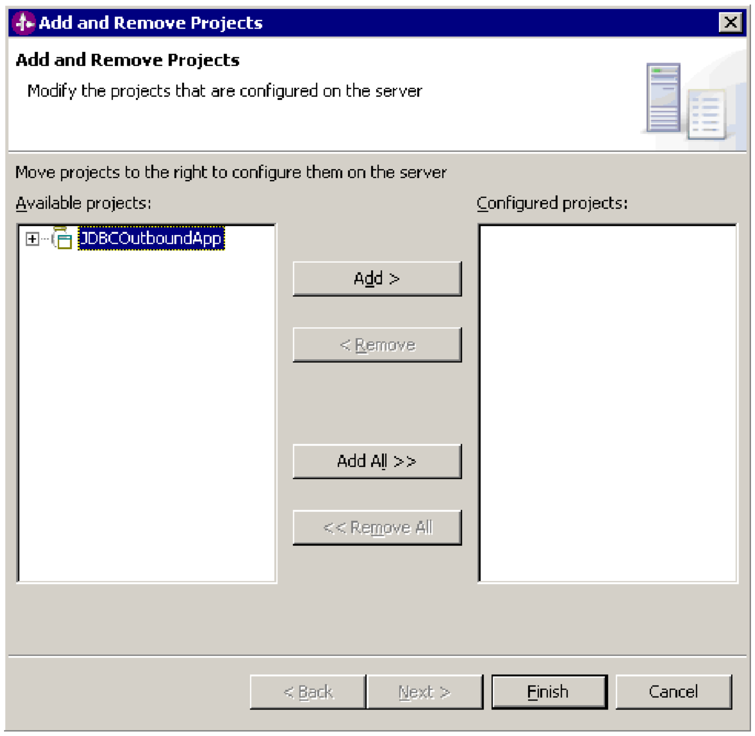


Figure 55. Select module to deploy on the server

5. Click **Finish**.

   This deploys the module (also called a project) on the server.

   The **Console** tab in the lower-right pane displays a status message log while the module is being added to the server. If any problems occur while you are adding the module, see "Troubleshooting the tutorial."

Result

After adding the module to the server, you can test that the record was created correctly in the database.

Testing the module

Test the module using the WebSphere Process Server test environment.

1. Right-click the **JDBCOutboundApp** module and select **Test** → **Test Module**.

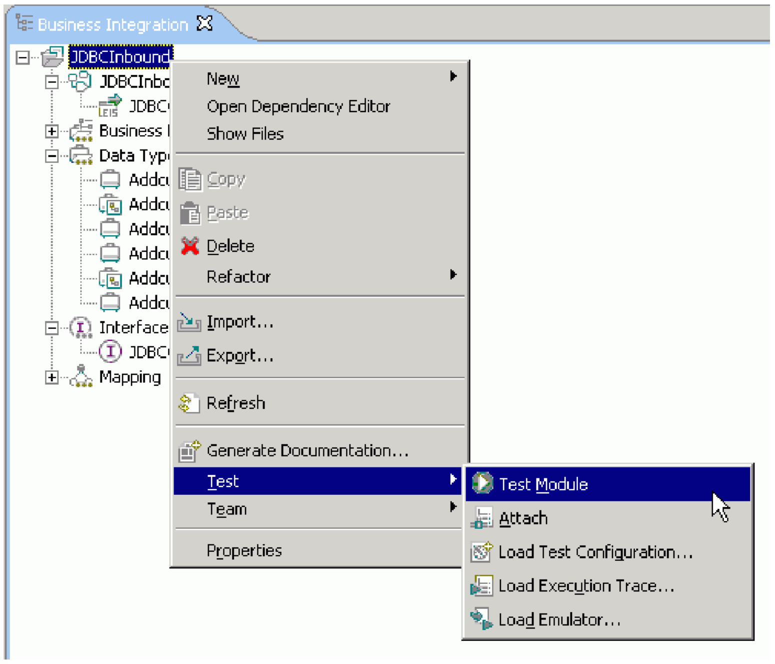
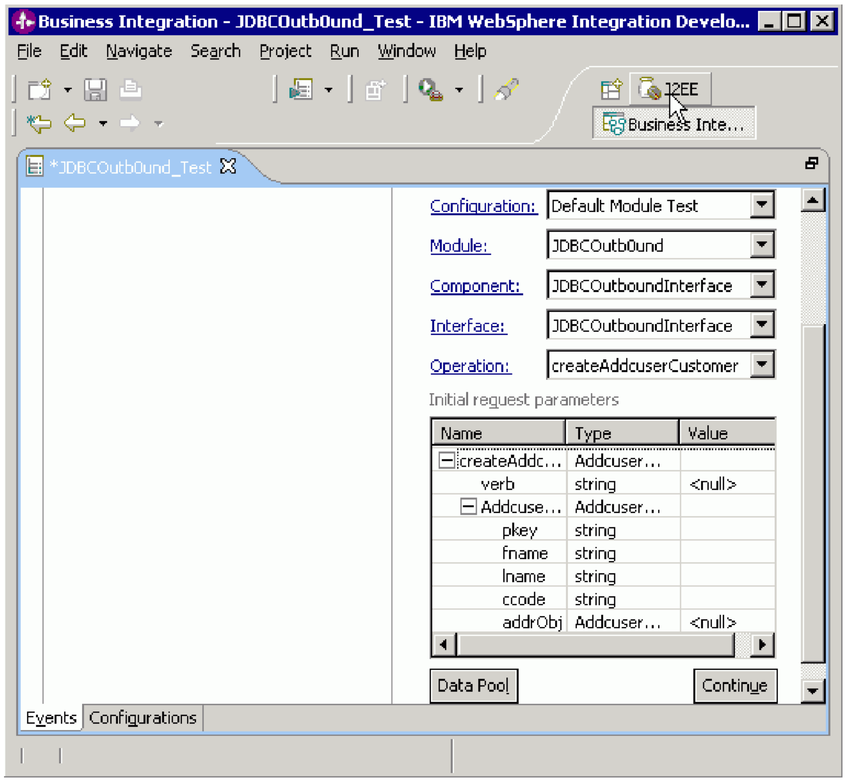


Figure 56. Test the module

2. In this scenario, you created a record in the database. To test that the record was created correctly. select **createAddcuserCustomer** in the **Operation** field.


Business Integration - JDBCOutbOund_Test - IBM WebSphere Integration Develo...
File Edit Navigate Search Project Run Window Help
J2EE
Business Inte...
*JDBCOutbOund_Test
Data Pool
Continue
Events Configurations

Configuration: Default Module Test
Module: JDBCOutbOund
Component: JDBCOutboundInterface
Interface: JDBCOutboundInterface
Operation: createAddcuserCustomer

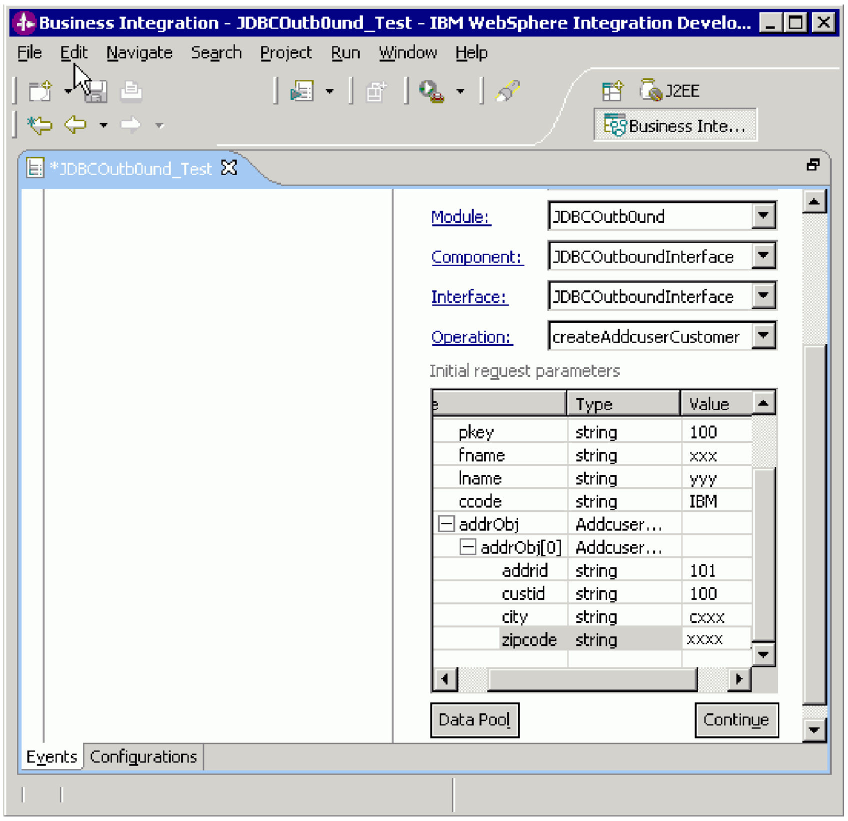
Initial request parameters

Name	Type	Value
⊟createAddc...	Addcuser...	
verb	string	<null>
⊟ Addcuse...	Addcuser...	
pkey	string	
fname	string	
lname	string	
ccode	string	
addrObj	Addcuser...	<null>

Figure 57. Select record you created

3. Select **Create** for the value of **verb**. Select the values for the parent business object **AddcuserCustomer**.

4. Right-click the **addrObj** attribute and select **Add Element** to create the child object.
5. Select the values for the child object, **AddcuserAddress**.


Business Integration - JDBCOutbOund_Test - IBM WebSphere Integration Develo...
File Edit Navigate Search Project Run Window Help
J2EE
Business Inte...
*JDBCOutbOund_Test
Events Configurations
Data Pool
Continue

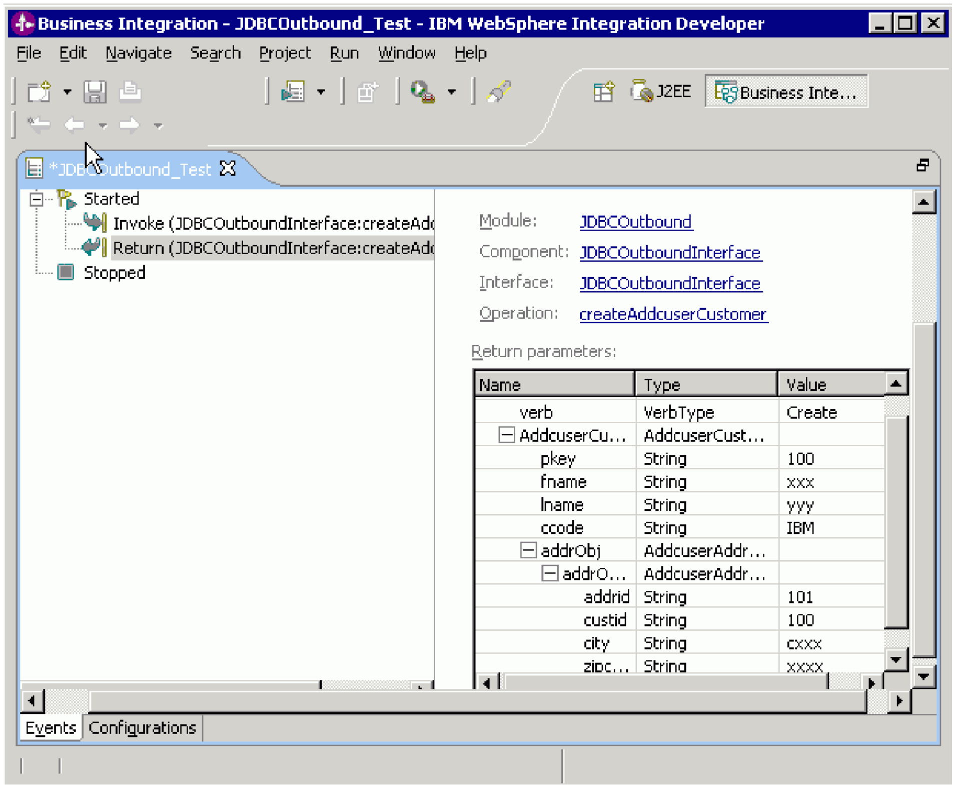
Module: JDBCOutbOund
Component: JDBCOutboundInterface
Interface: JDBCOutboundInterface
Operation: createAddcuserCustomer

Initial request parameters

e	Type	Value
pkey	string	100
fname	string	xxx
lname	string	yyy
ccode	string	IBM
addrObj	Addcuser...	
addrObj[0]	Addcuser...	
addrid	string	101
custid	string	100
city	string	cxxx
zipcode	string	xxxx

Figure 58. Select values for child object

6. Click **Continue** to run the outbound service.
7. In the Select Deployment Location window, click **WebSphere Process Server v6.0.2**.
8. Click **Finish**.
9. Check the output of the outbound service to ensure that the data in the database EIS matches the expected values; that is, that the new record was inserted in the database.


Business Integration - JDBCOutbound_Test - IBM WebSphere Integration Developer
File Edit Navigate Search Project Run Window Help
J2EE
Business Inte...
*JDBCOutbound_Test
Events
Configurations

Started
Invoke (JDBCOutboundInterface:createAdd
Return (JDBCOutboundInterface:createAdd
Stopped

Module: JDBCOutbound
Component: JDBCOutboundInterface
Interface: JDBCOutboundInterface
Operation: createAddcuserCustomer

Return parameters:

Name	Type	Value
verb	VerbType	Create
AddcuserCu...	AddcuserCust...	
pkey	String	100
fname	String	xxx
lname	String	yyy
ccode	String	IBM
addrObj	AddcuserAddr...	
addrO...	AddcuserAddr...	
addrid	String	101
custid	String	100
city	String	cxxx
zipc...	String	xxxx

Figure 59. Check results of outbound service

Troubleshooting the tutorial

When using the tutorial, you need to take action if the adapter fails to start up or if the adapter fails during processing.

1. The adapter could fail to start up. If this happens, see the adapter message status log file to find the cause of the failure.

   Common reasons why the adapter would fail to start up are listed here.

Table 26. Common reasons for adapter start-up failure

Error	Cause
Driver class does not exist.	The database driver JAR file does not exist in the runtime's lib folder.
Logon error; invalid username/password.	The authentication alias does not have the correct username or password for connecting to the database.

2. The adapter could fail during processing. If this happens, see the adapter message status log file to find the cause of the failure.

   Common reasons why the adapter could fail during processing are listed here.

Table 27. Common reasons for adapter failure during processing

Error	Cause
Primary key does not exist.	The table does not have a primary key defined; thus, the PrimaryKey application-specific information on the business object is not set to `true`.

Table 27. Common reasons for adapter failure during processing (continued)

Error	**Cause**
A record with the primary key already exists.	The record already exists in the database. Insert a record with a primary key that does not exist in the database.

Chapter 12. Viewing the sample adapter artifacts

To view the sample artifacts for each tutorial, import into IBM WebSphere Integration Developer the quick start reference files included with the adapter. Note that these artifacts are for reference only. They probably will not execute in your enterprise information system environment. If you have not stepped through the tutorials, you can still use the reference files to view examples of correctly-generated artifacts before you create your own.

Before you begin

Locate the quick start reference files in the referencefiles subdirectory of the samples directory. There is a project interchange zip file for each quick start tutorial. For instance, Tutorial1.zip is for quick start tutorial 1.

Important: Do not modify or use the artifacts provided in the quick start reference files. They are provided exclusively for viewing.

Reference files do not include third-party libraries. When imported into IBM WebSphere Integration Developer, the reference files might generate compilation error messages because dependent libraries are missing. The artifacts in the reference files may not be compatible with the enterprise information system (EIS) you are using. They vary based on EIS version and configuration.

About this task

Import the quick start reference files into WebSphere Integration Developer to view sample artifacts associated with each quick start tutorial.

How to perform this task

1. In the Business Integration perspective of WebSphere Integration Developer, click **File** → **Import**.
2. In the Import window, select **Project Interchange** and click **Next**.
3. Select the project interchange file containing the tutorial artifacts you want to view.
4. Import all the projects in the project interchange file by clicking **Select All** .
5. Click **Finish**.

Result

A business integration module is created with the following artifacts:
- Service import and export definitions
- Business objects (service data objects)
- Interfaces.

Chapter 13. Reference

Detailed information about adapter properties (enterprise service discovery connection properties, object selection properties, metadata selection properties, and adapter configuration properties), messages, and related product information is provided for your reference. Example import, export, and WSDL files are also provided. Instructions describe how to add JAR files to WebSphere Integration Developer v6.0.1.1 and earlier.

Enterprise service discovery connection properties

The enterprise service discovery process requires these properties to connect to the enterprise information system (EIS) for discovery and for creating the service description.

Using these properties, the discovery service prepares a metadata tree to be displayed for object selection and navigation.

The enterprise service discovery wizard uses the bidirectional connection properties to apply the proper bidirectional transformation on the data passed to the enterprise information system.

When you run the enterprise service discovery wizard in WebSphere Integration Developer, specify the connection properties listed below.

Enterprise service discovery connection properties for the Adapter for JDBC

Property	Required	Globalized	Description
Database URL	Yes	Yes	The database URL to connect to. The URL may vary from driver to driver. Specify this property according to the driver instructions. This property is enabled for use with bidirectional languages.
JDBCDriverClass	Yes	No	The name of the database driver used to connect to the database.
Password	Yes	Yes	Password for the corresponding user name. This property is enabled for use with bidirectional languages.
Prefix	No	Yes	The prefix to be added to the name of the business object. This property is enabled for use with bidirectional languages.
UserName	Yes	Yes	User name for logging in to the database. This property is enabled for use with bidirectional languages.

Bidirectional connection properties

Property	Value	Letter Position	Description
BiDi OrderingSchema	Implicit Visual	1	Logical (Implicit). Implicit is the default. Visual
BiDi Direction	LTR RTL contextual_LTR contextual_RTL	2	Left to Right. LTR (L) is the default. Right to Left Contextual Left to Right Contextual Right to Left
BiDi SymmetricSwapping	Yes No	3	Symmetric swapping is on. Yes (Y) is the default. Symmetric swapping is off
BiDi Shaping	nominal shaped initial middle final isolated	4	Nominal (N). This is the default. Letter shaped according to its position. Letter is in its initial shape. Letter is in its middle shape. Letter is in its final shape. Letter is in its isolated shape.
BiDi NumericShaping	nominal national contextual	5	Nominal (N). This is the default. Hindi (National) Contextual

Object selection properties

The enterprise service discovery wizard uses the Filter and Node properties during the business object selection process. Setting these properties narrows the list of schemas, nodes, or business objects displayed in the tree structure. After object selection, set properties to configure the business objects.

Filter and node properties

During business object discovery, you can specify Filter and Node properties if you want to narrow the list of schemas, node types, or database objects displayed in the tree; otherwise, all schemas, node types, or database objects are displayed. You can also specify properties for generating business objects from specified select statements.

To filter the schemas or node types, set application-specific information for the business object, or generate business objects from specified select statements, use the properties listed in the table below.

Table 28. Filter properties

Filter property	Type	Description	Default value
DefineASIs	Boolean	If `true`, when you select an object for generation, you are prompted to set the application-specific information (ASI) values for the business object. Select the check box to set this property to true.	false

Table 28. Filter properties (continued)

Filter property	Type	Description	Default value
QueryBO	Boolean	If `true`, you can specify a select statement to generate a query business object. Select the check box to set this property to true.	false
QueryBOCount	Integer	The maximum number of query business objects that can be generated at one time. This property is enabled only when the QueryBO property is set to `true.`	1
SchemaNameFilter	String	String used to filter schemas. Only those schemas that start with the string you specify are displayed. If this property is not set, all schemas are displayed.	None
Types	String; Multiple value	Determines which database object type nodes–tables, views, stored procedures, or synonyms/nicknames–to display under all the schemas listed. Values are set by enterprise service discovery. Multiple types may be selected.	None

To filter database objects, use the property listed in the table below.

Table 29. Node property

Node property	Type	Description	Default value
ObjectNameFilter	String	String used to filter database objects. Only those database objects that start with the text you specify are displayed. If no ObjectNameFilter is specified, all objects are displayed.	None

Metadata selection properties

After you have selected database objects, you need to set values for the metadata selection properties. The enterprise service discovery wizard queries for the metadata selection properties after confirming the list of selected database objects from which to generate business objects.

To set metadata values, use the properties listed in the table below.

Table 30. Metadata selection properties

Property	Type	Description
BOLocation	String	The path to the location where the generated .xsd files are to be stored. Will be set as RelativePath on DataDescription.
MaxRecords	Integer	The maximum number of records to retrieve for a RetrieveALL operation. This value is populated in the instance of InteractionSpec in the OutboundServiceDescription. The default value is 100.
NameSpace	String	This value is prepended to the business object name to keep the business object name logically separated. For example: `http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/Schema1Customer` The property is initially set to the default NameSpace for all business objects, which is `http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc`
Operations	String	Defaults to a list of the operations supported by the adapter for the selected ServiceType. Select from these operations. The specified operations are set for all business objects being generated. Query business objects support only RetrieveAll.

Table 30. Metadata selection properties (continued)

Property	Type	Description
ServiceType	String	Either inbound or outbound is chosen. The value is set by enterprise service discovery. Query business objects and stored procedure business objects support only the outbound service type.

Adapter configuration properties

This group of properties contains attributes used by the adapter to set up a communication channel to a specific database application. The properties include resource adapter, managed (J2C) connection factory, and activation specification properties.

Resource adapter properties

Resource adapter properties consist of logging and tracing, bidirectional language support, and activities specific to the adapter, such as the default configuration properties of the adapter. You configure these properties using the enterprise service discovery wizard or the administrative console of the server.

When you configure the adapter, specify the resource adapter properties listed in the tables below. The bidirectional resource adapter properties that need to be configured when bidirectional text transformation has been activated are described in a separate table in this section.

Table 31. Resource adapter properties for the Adapter for JDBC

Property	Type	Description	Globalized	Default value
BONameSpace	String	Namespace for the business object definitions to be used by this adapter. This value should be taken from the value you provided during the metadata discovery process. This property is required.	No	None
DatabaseVendor	String	Specifies which RDBMS the adapter uses for special processing. Set the value to `DB2`, `Oracle`, or `MSSQLServer` if using an IBM DB2, Oracle, or Microsoft SQL Server database. If using a different database, set the value to the name of that database. If using a non-default database, ensure that the proper driver is loaded. If this property is set to `Others`, the adapter determines which database to use by locating the driver. A value is required for the adapter to process successfully.	No	None
enableHASupport	String	When the enableHASupport property is set to `true`, only one of the replicated adapter instances actively polls for events while other instances are in standby mode. If the enableHASupport property is set to `false`, all of the adapter instances replicated on cluster members actively poll for events. This may result in event duplication. Do *not* change the value of enableHASupport to false for single server environments.		true

Table 31. Resource adapter properties for the Adapter for JDBC (continued)

Property	Type	Description	Globalized	Default value
LogFileName	String	The full path of the log file. This property is required.	No	None
LogNumberOfFiles	Integer	The number of log files to use. When a log file reaches its maximum size, another log file is started.	No	1
LogMaxFileSize	Integer	Size of the log files in kilobytes. If no value is specified, the files have no maximum size.	No	None
PingQuery	String	SQL query used to test valid connection to the database.	No	None
QueryTimeOut	Integer	This specifies the maximum number of seconds a query can take for all SQL statements. If a value is not specified, no timeout is set on the query. If the query takes longer than the number of seconds specified, the database produces an SQL exception that is captured. The associated message is logged in the log file.	Yes	None
ReturnDummyBO ForSP	Boolean	Used to return output parameters even when the result set is empty. In the case of RetrieveSP, a result set is returned. If the result set is empty, no business objects are created, and the output parameters returned by the procedure call cannot be retrieved. If this property value is `true`, a dummy business object with values from output and input/output parameters populated in the corresponding attributes is returned.	No	false
TraceFileName	String	The full path to the trace file. This property is required.	No	None
TraceNumberOfFiles	Integer	The number of trace files to use. When a trace file reaches its maximum size, another trace file is started.	No	1
TraceFileSizeMax	Integer	Size of the trace files in kilobytes. If no value is specified, the files have no maximum size.	No	None

The following table describes Resource adapter properties that need to be configured only when bidirectional text transformation has been activated.

Bidirectional resource adapter properties for the Adapter for JDBC

Property	Type	Description
BiDiContextEIS	String	Defines bidirectional format for content data in all of the business object runtime instances.
BiDiContextMetadata	String	Defines bidirectional format for metadata or configuration properties stored in the deployment descriptor.

Property	Type	Description
BiDiContextSkip	Boolean	Flag that controls the invocation of bidirectional support at the resource adapter level. If it equals `true`, the support will not be invoked. If it equals `false`, then the support will be invoked.
BiDiContextSpecialFormat	String	Specifies the category of properties subject to special treatment for all of the connector properties.
BiDiContextTurnBiDiOff	Boolean	Flag that controls the invocation of bidirectional transformation for the JDBC adapter. This parameter takes precedence over all the rest of the bidirectional parameters on all levels. In other words, if it is set to `true`, no bidirectional support is invoked, and no checks nor lookups are performed. If it is set to `false`, bidirectional support is invoked.

Managed (J2C) connection factory properties

Managed connection factory (MCF) configuration properties are used at run time to create an outbound connection instance with an enterprise information system. Once the MCF properties are created, they are stored in the deployment descriptor.

A J2C connection factory manages connection pooling. It provides configuration information for outbound connectivity to a single JDBC application instance from an application by way of the adapter.

When you configure the adapter, specify the properties listed in the tables below. The second and third tables in this section define the bidirectional managed (J2C) connection factory properties that need to be configured only when bidirectional text transformation has been activated.

Note: The enterprise service discovery wizard refers to these properties as managed connection properties, and the WebSphere Process Server refers to these as (J2C) connection factory properties.

Table 32. Managed (J2C) connection factory properties for the Adapter for JDBC

Property	Type	Description	Globalized	Bidirectional transformation supported
AutoCommit	Boolean	Value for AutoCommit to be set on the connection.	No	No
DatabaseURL	String	Database URL used to connect to the database.	Yes	Yes
DataSourceJNDIName		The JNDI data source name used to establish connection to the database. If the UserName and Password properties are also set, they will also be used when establishing the connection. If not, only this property will be used.	Yes	No

Table 32. Managed (J2C) connection factory properties for the Adapter for JDBC (continued)

Property	Type	Description	Globalized	Bidirectional transformation supported
JDBCDriverClass	String	JDBC driver class for the driver that is used to connect to the database.	No	No
Password	String	Password for the corresponding user name.	Yes	Yes
UserName	String	User name to log in to the enterprise information system.	Yes	Yes
XADataSourceName	String	XA datasource name to establish an XA connection to the database.	No	No
XADatabaseName	String	Database name used for the XA connection. This property must be set if you are using an IBM DB2 database.	Yes	Yes

Bidirectional managed J2C connection factory properties for the Adapter for JDBC

Property	Type	Description
BiDiContextEIS	String	Defines bidirectional format for content data in all of the business object runtime instances supported by the adapter for specific connection to the EIS.
BiDiContextMetadata	String	Defines bidirectional format for metadata or configuration properties for specific connection to the EIS.
BiDiContextSkip	Boolean	Flag that controls the invocation of bidirectional support at the resource adapter level. If it equals `true`, the support will not be invoked. If it equals `false`, then the support will be invoked.
BiDiContextSpecialFormat	String	Specifies the category of properties subject to special treatment for all of the connector properties of the specific connection to the EIS.
DatabaseURL BiDi properties		*DatabaseURLEIS, DatabaseURLSpecialFormat (default value is JDBC_URL_SQL), DatabaseURLSkip
Password BiDi properties		*PasswordEIS, PasswordSkip
UserName BiDi properties		*UserNameEIS, UserNameSkip
XADatabaseName BiDi properties		*XADatabaseNameEIS, XADatabaseNameSkip

*The three bidirectional (BiDi) properties associated with each BiDi-supported property. These properties are described in the following table.

BiDi property	Type	Description
BiDiContextCP<property_ Name>EIS	String	Defines bidirectional format for a specific connector configuration property for a specific connection to the EIS.
BiDiContextCP<property_ Name>SpecialFormat	String	Flag that controls the invocation of bidirectional support at the connector configuration property level. If it equals `true`, the support will not be invoked. It if equals `false`, then the support will be invoked.
BiDiContextCP<property_ Name>Skip	Boolean	Defines the special format for a specific connector property for a specific connection to the EIS.

Activation specification properties

Activation specification properties hold the inbound event processing configuration information for a message endpoint. They can be set through either the enterprise service discovery wizard or the WebSphere Application Server or WebSphere Enterprise Service Bus administrative console.

When you configure the adapter, specify the activation specification properties listed the tables below. A separate table describes the bidirectional activation specification properties that need to be configured only if bidirectional text transformation is being used.

Table 33. Activation specification properties for the Adapter for JDBC

Property	Type	Description	Globalized	Default value
AssuredOnceDelivery	Boolean	If this property is set to `true`, an xid value is set for each event in the event store. Each event is then delivered to its corresponding endpoint and subsequently deleted from the event table.	No	true
CustomDeleteQuery	String	The custom delete query that is run after each event is processed. This is enabled for use with bidirectional languages.	Yes	None
CustomEventQuery	String	The SQL query, stored procedure, or stored function for custom event processing. This query is run during each poll cycle when the EventQueryType is set to Dynamic. This is enabled for use with bidirectional languages.	Yes	None
CustomUpdateQuery	String	The custom update query that is run after each event is processed so that the same event does not get picked up for processing in the subsequent event cycle. This is enabled for use with bidirectional languages.	Yes	None
DatabaseURL	String	The driver-specific URL for creating a connection to the enterprise information system (EIS). This is enabled for use with bidirectional languages.	Yes	None

Table 33. Activation specification properties for the Adapter for JDBC (continued)

Property	Type	Description	Globalized	Default value
DataSource JNDIName	String	Name used by the adapter to establish the connection to the database. If UserName and Password are also set, they are also used when establishing the connection. If not, just the DataSourceJNDIName property is used.	Yes	None
Delivery Type	String	This property determines the order in which the events are published. The value is either `ORDERED` or `UNORDERED`.*Ordered* means events are published one at a time, while *unordered* means events are published all at once.	Yes	ORDERED
EventFilterType	String	The adapter can filter the events to be processed by business object type. EventFilterType has a comma delimited list of business object types, and only the types specified in the property are picked up for processing. If no value is specified for the property, no filter is applied, and all the events are picked up for processing.	Yes	null
EventOrderBy	String	The order in which events are retrieved and processed. Expected values are comma-separated column names of the event table. This property is enabled for use with bidirectional languages.	No	None
EventQueryType	String	Determines whether to use the standard event store or custom query. The valid values are `Standard`, for standard event store, and `Dynamic`, for custom event processing.	No	None
EventTableName	String	Table in the EIS that contains events generated by the EIS for inbound processing. This is enabled for use with bidirectional languages.	Yes	None
FilterFutureEvents	Boolean	If this property is set to `true`, the adapter filters events based on timestamp. The adapter compares the system time in each poll cycle to the timestamp on each event. If an event is set to occur in the future, it is not picked up for processing until that time.	No	false
JDBCDriverClass	String	JDBC driver class used to connect to the EIS.	No	None
Password	String	Password for authorizing the user to retrieve events from the EIS. This property is enabled for use with bidirectional languages.	Yes	None

Table 33. Activation specification properties for the Adapter for JDBC (continued)

Property	Type	Description	Globalized	Default value
PollPeriod	Integer that is equal to or greater than 0	The rate in milliseconds at which to poll the EIS event store for new inbound events. If `0`, the adapter will not wait between cycles. The poll cycle is established at a fixed rate, meaning that if an execution of the poll cycle is delayed (for example, the prior poll cycle takes longer than expected to complete) the next cycle will occur immediately to catch up. This is a required property.	Yes	500
PollQuantity	Integer that is greater than 0.	This property is used to determine the number of events to deliver to each endpoint per poll cycle. It is a required property.	Yes	None
RetryLimit	Integer	The number of times the adapter attempts to retry the connection to the EIS. A value of `0` (zero) causes the adapter to use an unlimited number of attempts.	No	0
RetryInterval	Integer	If this value is non-negative and the adapter encounters an error related to the inbound connection to the EIS, the adapter continues to re-attempt a connection after the specified interval until a connection is re-established. Values are in milliseconds.	No	60000 (1 minute)
SPAfterPoll	String	Any stored procedure that you want to be run after each poll cycle. It takes one input parameter for PollQuantity. This is enabled for use with bidirectional languages.	Yes	None
SPBeforePoll	String	Any stored procedure that you want to be run before the actual poll query is called. It takes one input parameter for PollQuantity. This is enabled for use with bidirectional languages.	Yes	None
StopPollingOnError	Boolean	If `true`, the adapter stops polling when it encounters an error during polling. If `false`, the adapter logs an exception when it encounters an error, and continues polling.	No	false
UserName	String	User name for logging into the EIS for inbound events. This is enabled for use with bidirectional languages.	Yes	None

Bidirectional activation specification properties for the Adapter for JDBC

Property	Type	Description
BiDiContextEIS	String	Defines bidirectional format for content data in all of the business object runtime instances supported by the adapter for specific connection to the EIS.

Property	Type	Description
BiDiContextMetadata	String	Defines the bidirectional format for metadata or configuration properties for a specific connection to the EIS.
BiDiContextSkip	Boolean	Flag that controls the invocation of bidirectional support at the resource adapter level. If it equals `true`, the support will not be invoked. If it equals `false`, then the support will be invoked.
BiDiContextSpecialFormat	String	Specifies the category of properties subject to special treatment for all of the connector properties of a specific connection to the EIS.
CustomDeleteQuery BiDi properties		*CustomDeleteQueryEIS CustomDeleteQuerySkip
CustomEventQuery BiDi properties		*CustomEventQueryEIS CustomEventQuerySkip
CustomUpdateQuery BiDi properties		*CustomUpdateQueryEIS CustomUpdateQuerySkip
DatabaseURL BiDi properties		*DatabaseURLEIS, DatabaseURLSpecialFormat (default value is JDBC_URL_SQL), DatabaseURLSkip
EventTableName BiDi properties		*EventTableNameEIS, EventTableNameSkip
EventOrderBy BiDi properties		*EventOrderByEIS, EventOrderBySkip
Password BiDi properties		*PasswordEIS, PasswordSkip
SPAfterPoll BiDi properties		*SPAfterPollEIS, SPAfterPollSkip
SPBeforePoll BiDi properties		*SPBeforePollEIS, SPBeforePollSkip
UserName BiDi properties		*UserNameEIS, UserNameSkip

*The three bidirectional (BiDi) properties associated with each BiDi-supported property. These properties are described in the following table.

BiDi property	Type	Description
BiDiContextCP<property_ Name>EIS	String	Defines bidirectional format for a specific connector configuration property for a specific connection to the EIS.
BiDiContectCP<property_ Name>SpecialFormat	String	Flag that controls the invocation of bidirectional support at the connector configuration property level. If it equals `true`, the support will not be invoked. If it equals `false`, then the support will be invoked.
BiDiContextCP<property_ Name>Skip	Boolean	Defines special format for a specific connector property for a specific connection to the EIS.

Important: Custom queries for CustomEventQuery, CustomUpdateQuery, and CustomDeleteQuery properties are supported for bidirectional transformation when they are in the form of stored procedures or stored functions. But, for standard SQL only the string constants in the SQL statement are supported.

Interaction specification property

The adapter uses Interaction specifications as the client interface. The Interaction specifications specify an operation name. The adapter extracts the name and performs the corresponding operation on the input business object instance.

Table 34. Interaction specification property for the Adapter for JDBC

Property	Type	Description	Globalized	Default Value
ResultSetLimit	Integer	Maximum number of result sets to return during a RetrieveAll operation. If the number of hits in the database exceeds the value of ResultSetLimit, the adapter returns the error MatchesExceededLimitException. The adapter uses this property to help avoid out-of-memory issues.	No	100

Example import, export, and WSDL files

Service Component Architecture (SCA) import and export files and Web Services Description Language (WSDL) files are artifacts produced during the enterprise service discovery process.

Examples are provided below of an export file and an import file, plus a corresponding WSDL file.

Example export (inbound) file

```
<?xml version="1.0" encoding="UTF-8"?>
<scdl:Export xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:_="http://JDBCEMD/inbound"
xmlns:eis="http://www.ibm.com/xmlns/prod/websphere/scdl/eis/6.0.0"
xmlns:scdl="http://www.ibm.com/xmlns/prod/websphere/scdl/6.0.0"
xmlns:wsdl="http://www.ibm.com/xmlns/prod/websphere/scdl/wsdl/6.0.0"
name="inbound/JDBCInboundInterface">
  <interfaces>
    <interface xsi:type="wsdl:WSDLPortType" portType="_:JDBCInboundInterface"/>
  </interfaces>
  <esbBinding xsi:type="eis:EISExportBinding"
dataBindingType="com.ibm.j2ca.extension.emd.runtime.WBIDataBindingImpl">
    <resourceAdapter name="JDBCEMDApp.IBM JDBC Adapter"
type="com.ibm.j2ca.jdbc.JDBCResourceAdapter">
      <properties/>
    </resourceAdapter>
    <connection type="com.ibm.j2ca.jdbc.inbound.JDBCActivationSpec"
selectorType="com.ibm.j2ca.extension.emd.runtime.WBIFunctionSelectorImpl">
      <properties>
        <BONamespace>http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc</BONamespace>
        <jdbcDriverClass>COM.ibm.db2.jdbc.app.DB2Driver</jdbcDriverClass>
        <databaseURL>jdbc:db2:somedb<databaseURL>
        <password>abcdefg</password>
        <userName>db2admin</userName>
      </properties>
    </connection>
    <methodBinding method="createDb2adminCustomer"
nativeMethod="emitCreateAfterImageDb2adminCustomer"/>
    <methodBinding method="updateDb2adminCustomer"
nativeMethod="emitUpdateAfterImageDb2adminCustomer"/>
```

```
    <methodBinding method="deleteDb2adminCustomer"
nativeMethod="emitDeleteAfterImageDb2adminCustomer"/>
  </esbBinding>
</scdl:Export>
```

Example import (outbound) service description

```
 <?xml version="1.0" encoding="UTF-8"?>
<scdl:Import xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:_="http://JDBCEMD/outbound"
xmlns:eis="http://www.ibm.com/xmlns/prod/websphere/scdl/eis/6.0.0"
xmlns:scdl="http://www.ibm.com/xmlns/prod/websphere/scdl/6.0.0"
xmlns:wsdl="http://www.ibm.com/xmlns/prod/websphere/scdl/wsdl/6.0.0"
name="outbound/JDBCOutboundInterface">
  <interfaces>
    <interface xsi:type="wsdl:WSDLPortType" portType="_:JDBCOutboundInterface"/>
  </interfaces>
  <esbBinding xsi:type="eis:EISImportBinding"
dataBindingType="com.ibm.j2ca.extension.emd.runtime.WBIDataBindingImpl">
    <resourceAdapter name="JDBCEMDApp.IBM JDBC Adapter"
type="com.ibm.j2ca.jdbc.JDBCResourceAdapter">
      <properties/>
    </resourceAdapter>
    <connection type="com.ibm.j2ca.jdbc.JDBCManagedConnectionFactory"
interactionType="com.ibm.j2ca.jdbc.JDBCInteractionSpec">
      <properties>
        <databaseURL>jdbc:db2:somedb</databaseURL>
        <jdbcDriverClass>COM.ibm.db2.jdbc.app.DB2Driver</jdbcDriverClass>
        <password>abcdefg</password>
        <userName>db2admin</userName>
      </properties>
    </connection>
    <methodBinding method="createDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Create</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="updateDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Update</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="deleteDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Delete</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="retrieveDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Retrieve</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="retrieveallDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>RetrieveAll</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="applychangesDb2adminCustomer">
      <interaction>
        <properties>
        <functionName>ApplyChanges</functionName>
      </properties>
    </interaction>
  </methodBinding>
 </esbBinding>
</scdl:Import>
```

Example WSDL file

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="CUSTOMER"
        targetNamespace="http://test/j2c/jdbc/customer"
        xmlns:tns="http://test/j2c/jdbc/customer"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:datans="http://test/j2c/jdbc/customer">
        <wsdl:types>
                <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
                        <xsd:import namespace="http://test/j2c/jdbc/customer"
                                schemaLocation="CUSTOMER.xsd">
                        </xsd:import>
                </xsd:schema>
        </wsdl:types>
        <wsdl:message name="CUSTOMERRequest">
                <wsdl:part name="request" element="datans:CUSTOMER"></wsdl:part>
        </wsdl:message>
        <wsdl:portType name="Customer">
                <wsdl:operation name="updateCustomer">
                        <wsdl:input message="tns:CustomerRequest"></wsdl:input>
                        <wsdl:output message="tns:CustomerRequest"></wsdl:output>
                </wsdl:operation>
                <wsdl:operation name="createCustomer">
                        <wsdl:input message="tns:CustomerRequest"></wsdl:input>
                        <wsdl:output message="tns:CustomerRequest"></wsdl:output>
                </wsdl:operation>
                <wsdl:operation name="retrieveCustomer">
                        <wsdl:input message="tns:CustomerRequest"></wsdl:input>
                        <wsdl:output message="tns:CustomerRequest"></wsdl:output>
                </wsdl:operation>
        </wsdl:portType>
</wsdl:definitions>
```

Adding jar files to WebSphere Integration Developer versions 6.0.1.1 and earlier

If you are using WebSphere Integration Developer version 6.0.1.1 or earlier, you must manually add three jar files to the classpath of the connector project.

You must have installed the adapter and all of the adapter prerequisites before the jar files can be added to the connector project in WebSphere Integration Developer.

1. Open WebSphere Integration Developer.
2. In J2EE perspective, right-click the connector project and select **Properties**.
3. Select **Java Build Path** and click **Add External Jars**.
4. Select your WebSphere Process Server or Enterprise Server Bus Install/lib folder and select ffdcSupport.jar, aspectjrt.jar and icu4j_3_2.jar.
5. Click **Open** and then **OK**.

Messages

The messages issued by IBM WebSphere Adapters are documented in the WebSphere Adapters, version 6.0.2 information center.

You can view the adapter messages at the following link: WebSphere Adapters messages..

Related product information

The following links, information centers, Redbooks, and Web pages contain related information for the IBM WebSphere Adapter for JDBC.

Additional information you might need

Table 35. WebSphere Adapters information you might need

Information	How to find it
How to edit business objects using the Business Object Editor	In the IBM WebSphere Business Process Management information center, which includes documentation for WebSphere Integration Developer, search for the topic, ″Editing Business Objects.″
How to uninstall a deployed adapter	On the WebSphere Application Server library page, open the information center for your version of WebSphere Application Server and search for the topic, ″Uninstalling applications.″

Information for related products

- WebSphere Adapters, Version 6.0
- WebSphere Business Integration Adapters
- WebSphere Integration Developer
- WebSphere Process Server
- WebSphere Enterprise Service Bus
- WebSphere Application Server

Redbooks

- WebSphere Adapter Development Redbook
- WebSphere Redbooks domain

developerWorks® resources

- WebSphere Adapter Toolkit
- WebSphere business integration zone

Support and assistance

- WebSphere Adapters product support
- WebSphere Adapters technotes - in the **Additional search terms** field, specify the name of the adapter and click **Go**.

Chapter 14. Glossary

The glossary of terms for IBM WebSphere Adapters is included in the WebSphere Adapters, version 6.0.2 information center.

You can view it at the following link: WebSphere Adapters glossary.

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:

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

Index

Printed in USA