

webMethods EntireX

Calling Integration Server from Natural

Version 10.9

April 2023

This document applies to webMethods EntireX Version 10.9 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1997-2023 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at <http://softwareag.com/licenses/> and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: EXX-EEXXSCENARIO-IS2NAT-109-20230403

Table of Contents

1 About this Documentation	1
Document Conventions	2
Online Information and Support	2
Data Protection	3
2 Calling Integration Server from Natural	5
Introduction	6
What do I need to Install for this Scenario?	8
Task 1: Generate Integration Server Connections and Listeners	8
Task 2: Generate Client Interface Objects and Build Client Application	17
Task 3: Execute the Call from Natural to Integration Server	19

1

About this Documentation

■ Document Conventions	2
■ Online Information and Support	2
■ Data Protection	3

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Monospace font	Identifies service names and locations in the format <i>folder.subfolder.service</i> , APIs, Java classes, methods, properties.
<i>Italic</i>	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
Monospace font	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the symbol.
[]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at <https://documentation.softwareag.com>.

In addition, you can also access the cloud product documentation via <https://www.software-ag.cloud>. Navigate to the desired product and then, depending on your solution, go to “Developer Center”, “User Center” or “Documentation”.

Product Training

You can find helpful product training material on our Learning Portal at <https://knowledge.softwareag.com>.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at <https://tech-community.softwareag.com>. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at <https://github.com/softwareag> and <https://hub.docker.com/publishers/softwareag> and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at <https://empower.softwareag.com>. Many services on this portal require that you have an account. If you do not yet have one, you can request it at <https://empower.softwareag.com/register>. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

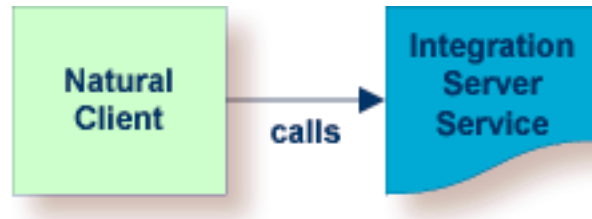
Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 **Calling Integration Server from Natural**

■ Introduction	6
■ What do I need to Install for this Scenario?	8
■ Task 1: Generate Integration Server Connections and Listeners	8
■ Task 2: Generate Client Interface Objects and Build Client Application	17
■ Task 3: Execute the Call from Natural to Integration Server	19

Scenario: “I have an Integration Server service and want to call this from a Natural application.”

This scenario uses the tools IDL Extractor for Integration Server and Natural Wrapper of the Designer.



See also *Common Integration Scenarios* for other common user scenarios that can be handled by EntireX.

Introduction

To call the Integration Server from Natural, take an existing Integration Server service ❶ and generate the integration logic ❷ to call it from a Natural application ❸, as shown below.



- ❶ Read package from Integration Server and generate Integration Server connection and listeners. See *Using the IDL Extractor for Integration Server*.
- ❷ Generate client interface objects for Natural client application and Natural test programs. See *Natural Wrapper*.
- ❸ Execute call from Natural client to Integration Server listener. See *Sample Generation Result for the Client Side*.

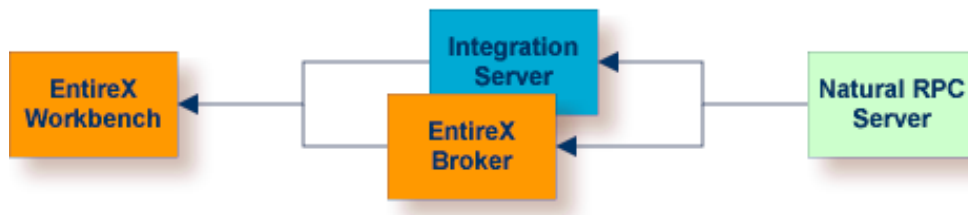
This scenario makes the following important assumptions:

For Task 1:

- You have a working Integration Server Service that you want to call from Natural.

For Task 2:

- You have a running Natural RPC Server. Generating Natural code requires a running Natural RPC Server and either EntireX Broker or Integration Server.



See your Natural documentation for setting up the Natural RPC Server.

For Task 3:

- You have an Integration Server with the EntireX Adapter installed.
- You can call the Integration Server service at runtime using different methods:
 - For the EntireX RPC connection method you need EntireX Broker on one of the supported platforms: z/OS | Linux | Windows | BS2000.



- For the EntireX Direct RPC connection method there are no additional prerequisites. See also *Direct RPC*.



What do I need to Install for this Scenario?

- You have Software AG Designer installed with EntireX plugins and Service Development plugins.
 - For EntireX plugins, see *Designer > EntireX Designer* under *EntireX Installation Packages* in the General Installation documentation.
 - For Service Development plugins, refer to the Integration Server documentation.
- You have an Integration Server with EntireX Adapter installed. See *Adapters > EntireX Adapter* under *EntireX Installation Packages* in the General Installation documentation.
- Optionally, for RPC Connection method (Task 3), you have an EntireX Broker installed.
 - For Linux and Windows, see *EntireX Broker* under *EntireX Installation Packages* in the General Installation documentation.
 - For z/OS, see *Installing EntireX Broker under z/OS* in the z/OS Installation documentation
- You have Natural and a Natural RPC Server installed. See your Natural documentation for information on installing Natural and setting up a Natural RPC Server.

Task 1: Generate Integration Server Connections and Listeners

- [Introduction](#)
- [Sample webMethods IS Service](#)
- [Extracting the Interface of the webMethods IS Service](#)
- [Creating Listener Objects in Integration Server](#)
- [Viewing the Extracted IDL](#)
- [Testing the Extraction Results \(Optional\)](#)

Introduction

Follow the instructions for extracting a webMethods Integration Server (IS) service under *Using the IDL Extractor for Integration Server*.

This process creates the following EntireX metafiles:

- an IDL file, containing definitions of the interface between client and server; see *Software AG IDL File* in the IDL Editor documentation in the IDL Editor documentation
- a webMethods IS Connection
- a webMethods IS Listener, depending on connection type. See *Listeners* in the EntireX Adapter documentation.

Connection types are described under [Creating Listener Objects in Integration Server](#).

Sample webMethods IS Service

Imagine that a new order management system is being developed with webMethods Integration Server in your organization. A package *OrderManagement* with flow service is provided. From your existing Natural application you would like to call the flow service *placeOrder* on the IS.

The screenshot shows the 'Results' window in webMethods Integration Server. The top bar indicates the execution of the service `[localhost:5555] OrderManagement.service:placeOrder`. The main area is divided into two panes. The left pane shows the 'Request' and 'Response' data in a table format. The right pane shows a tree view of the service structure, with the 'placeOrder' service highlighted.

Name	Value
Request	
OrderItemID	45
OrderQuantity	18
OrderCustomerID	234
Response	
OrderItemName	tennis racquet
OrderUnitPrice	79.89 GBP
OrderTotal	1438.02 GBP
OrderCustomerAddress	
Name	The All England Lawn Tennis Club (Championships) Limited
Street	Church Road
City	Wimbledon London
ZipCode	SW19 5AE
CountryCode	UK
OrderShippingDate	Your order will be shipped on Mo, Jul 4, 2016.
OrderConfirmation	Your order is confirmed.

The tree view on the right shows the following structure:

- OrderManagement
 - OrderManagement
 - documents
 - Address
 - Request
 - Response
 - service
 - placeOrder
 - WmEntireX
 - WmFlatFile
 - WmISExtDC

The following input is required:

- ID of the item to be ordered (*OrderItemID*)
- a quantity (*OrderQuantity*)
- customer ID *OrderCustomerID*)

The following output is returned:

- item name (*OrderItemName*)
- unit and total price (*OrderUnitPrice* and *OrderTotal*)
- etc.

Extracting the Interface of the webMethods IS Service

First create a project in Designer. This project is the container into which all extracted and generated EntireX artifacts are placed. Name this project *Natural placeOrder client*.

➤ To start the IDL Extractor for Integration Server

- 1 Switch to the EntireX perspective.
- 2 Invoke the IDL Extractor for Integration Server, which is a New Wizard in Eclipse. From the **File** menu, choose **New**. Select IDL Extractor for webMethods IS in the following page.

Or:

Choose **New** from the toolbar or context menu of a view showing resources.

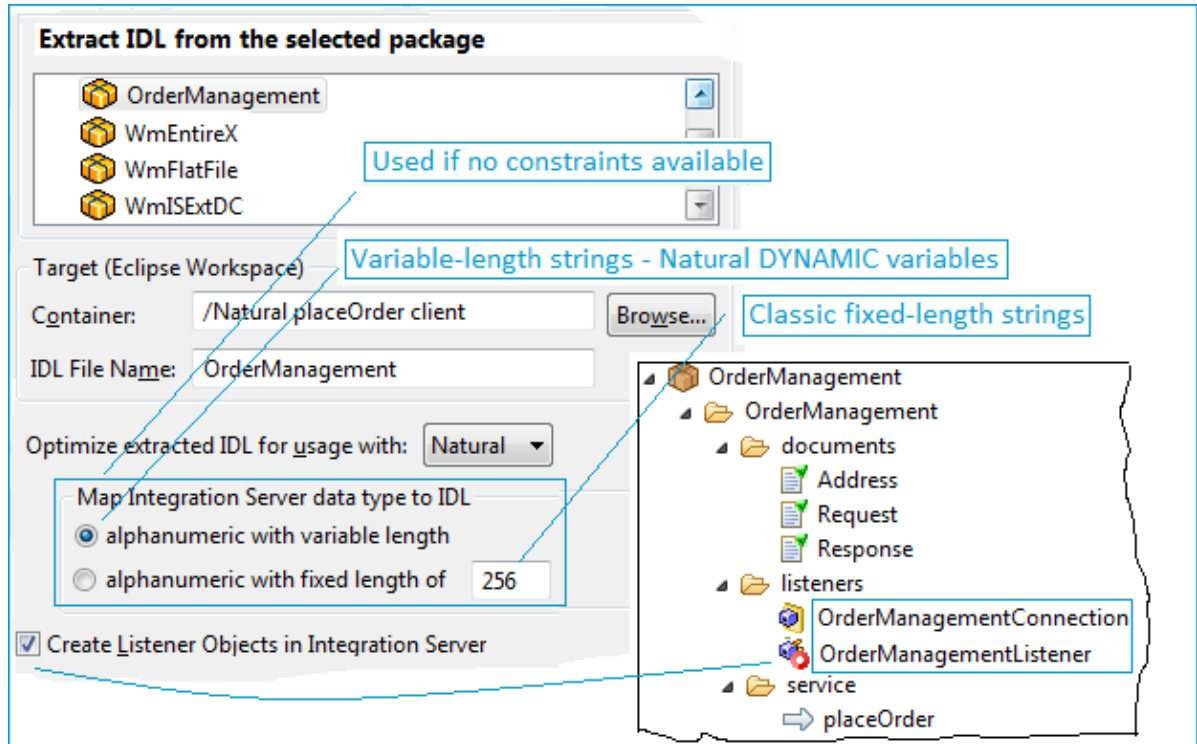
Or:

Press **Ctrl-N** to start the selection of the New Wizards.

- 3 If you are using the wizard for the first time without any predefined Integration Server connections, enter the TCP/IP address of the webMethods IS to extract from. Otherwise select the connection to the Integration Server.
- 4 Press **Next**.

➤ To extract the IDL from the selected package

- 1 Under **Source (Integration Server)**, select the package you want to extract from.



- 2 For **Target (Eclipse Workspace)**, specify project *Natural placeOrder client*. Extraction results will be placed in this container. You can keep the **IDL File Name** as it has been derived from the package name by default. Since Natural is your desired endpoint, select **Optimize extracted IDL for usage with: Natural**.

With **Map Integration Server data type to IDL** you define the default for webMethods IS variable length types with no specified constraints. For historical reasons, you can map them to two types of string:

- Natural variable-length strings, so-called Natural DYNAMIC variables
- classic Natural fixed-length strings

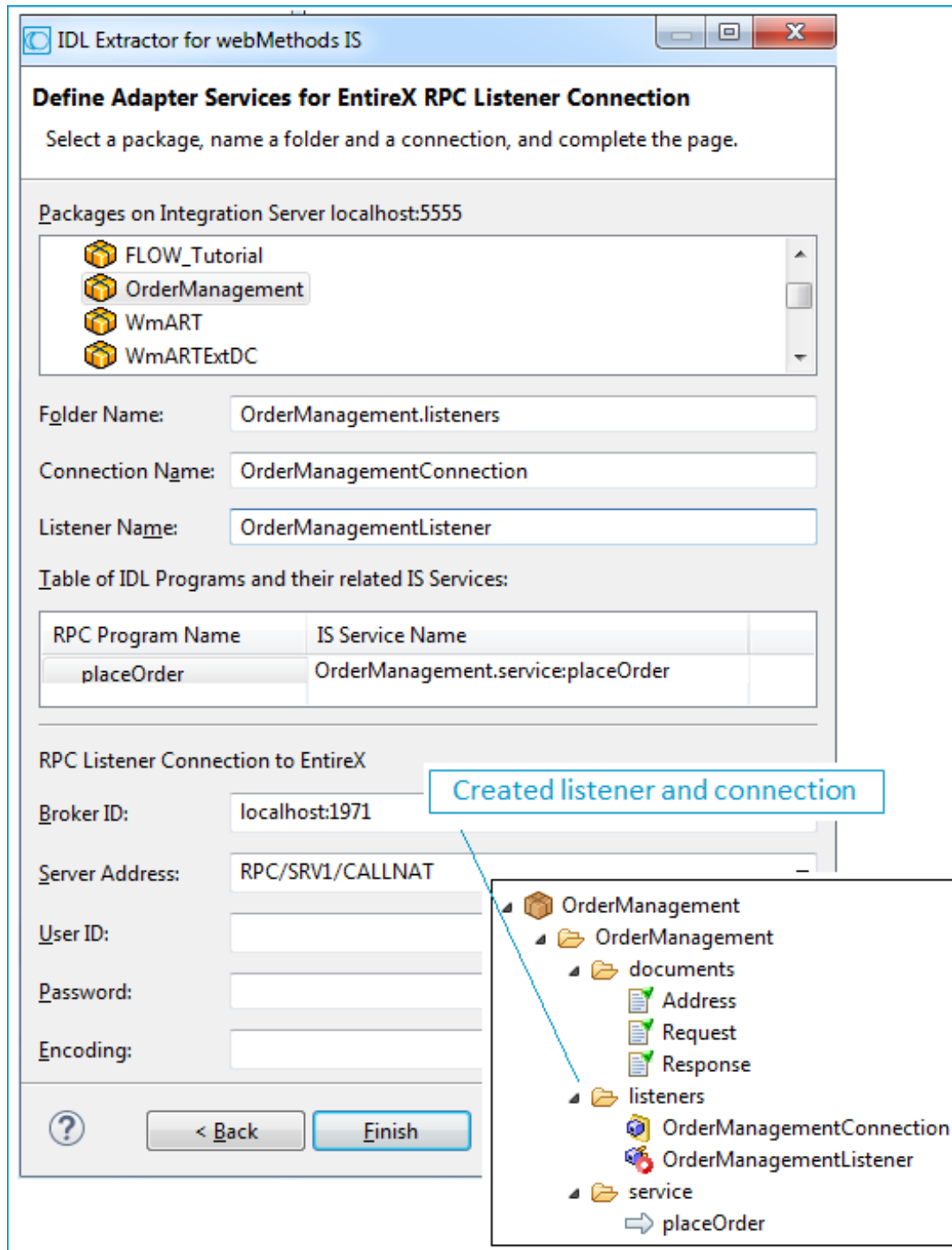
You can see the effect of this default under [Viewing the Extracted IDL](#).

- 3 A listener is required for calling webMethods IS from an RPC client, so check **Create Listener Objects in Integration Server**.
- 4 Press **Next**.

Creating Listener Objects in Integration Server

➤ To create Listener Objects in Integration Server

- 1 Select a Listener connection type used for inbound connection to the Integration Server.
 - An EntireX RPC Listener connection is the standard way and is always available.
 - If your interface contains only In parameters, an EntireX Reliable RPC Listener connection is available.
 - If it is enabled by the license for the webMethods Integration Server:
 - an EntireX Direct RPC Listener connection is available
 - if your interfaces contain only In parameters, an EntireX Direct Reliable RPC Listener connection is available



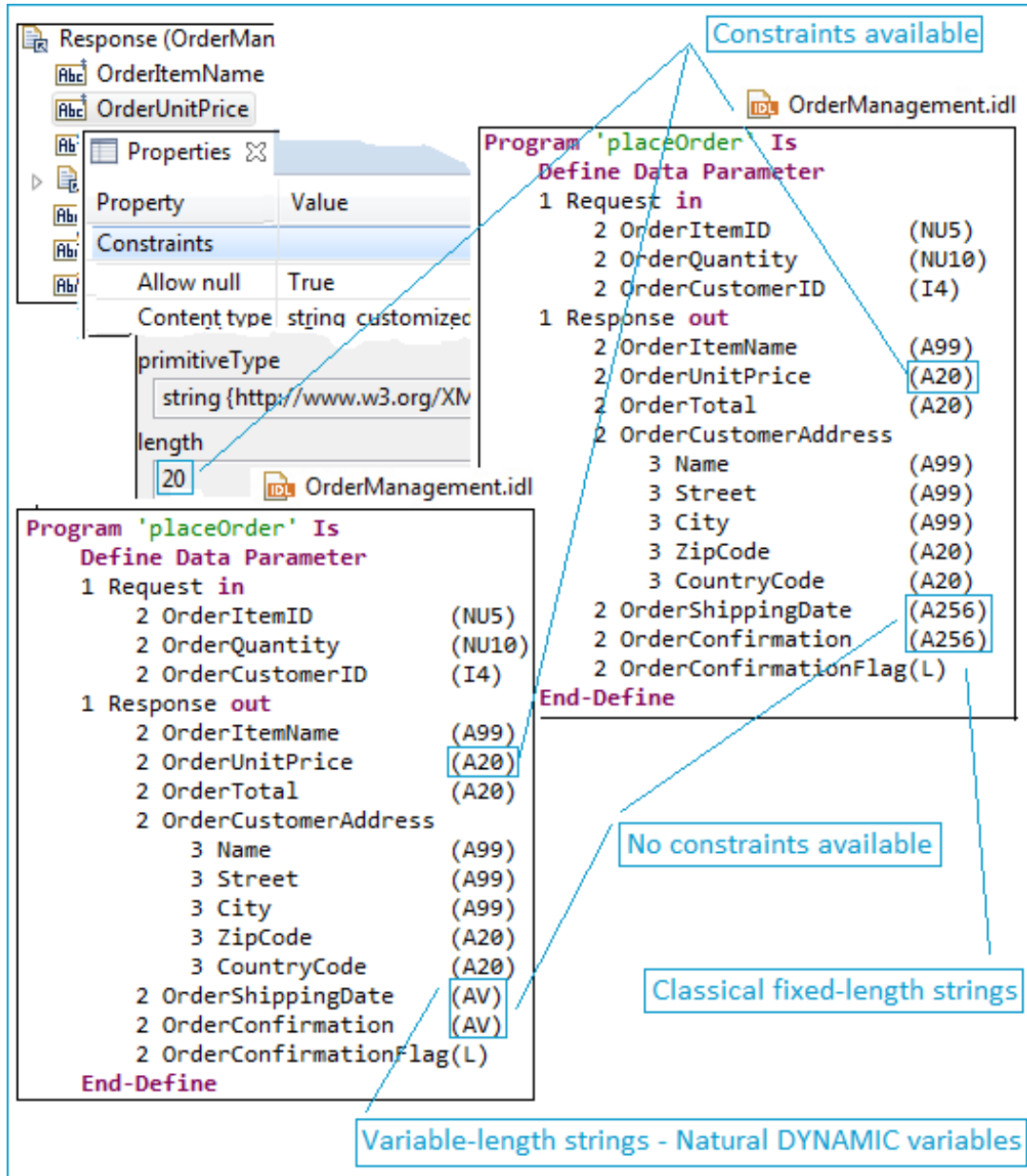
- On this page, you set the properties for the listener objects to create; these objects will wait for the incoming Natural requests. Select the package and specify the folder name into which the listener is generated. If the folder does not exist, it will be created automatically.

- 3 Keep the defaults given for **RPC Listener Connection to EntireX**. The **Broker ID** is a TCP/IP or SSL/TLS address and **Server Address** is an EntireX-specific namespace to locate a target server (here: *OrderManagementListener*).
- 4 When creating a connection, a package dependency is added such that the selected package depends on webMethods EntireX (the package WmEntireX) with the version currently used.
- 5 Press **Finish**. The listener and the connection appear in the package navigator. For the package navigator, switch to the service development perspective by choosing **Window > Perspective > Open Perspective > Service Development**.

Viewing the Extracted IDL

➤ To view the extracted the IDL

- The Software AG IDL is saved in the target Eclipse workspace as selected: container *Natural placeOrder client* and IDL file *OrderManagement.idl*:



The IDL file is the central artifact from which Natural code is generated later in [Task 2: Generate Client Interface Objects and Build Client Application](#). Extracted Software IDL data and resulting Natural types in the generated client interface objects depend on constraints and on the extraction settings under **Map Integration Server data type to IDL**, option **alphanumeric with variable length** and **alphanumeric with fixed length**. See [Extracting the Interface of the webMethods IS Service](#).

It is best if the Natural types match your data. This means:

- Numeric data should be mapped to numeric Natural types of the appropriate range. In our example this is the case for *OrderItemID*, *OrderQuantity* and *OrderCustomerID*.

- For string data where the length is known, set a length constraint too. This is the case for *OrderUnitPrice*, *OrderTotal*, *ZipCode* and *CountryCode*. For fixed-length short fields (1-4 bytes) this shortens the message length and improves performance.
- We recommend you set the option **alphanumeric with variable length** (see [Extracting the Interface of the webMethods IS Service](#)) as the default for string data where the length is unknown and no length constraint could be set as for IS parameter *OrderShippingDate* and *OrderConfirmationFlag*. This results in IDL type AV (see above, left IDL example) and - last but not least - in Natural type A DYNAMIC in the generated client interface objects. See [Task 2: Generate Client Interface Objects and Build Client Application](#). You have more freedom and no limitations on the length of the string being transferred.

The option **alphanumeric with fixed length** is there for historical reasons. It results in a classic fixed-length Natural type A with length for IS string parameter without constraints, in our example A256 as Natural and IDL data type (see IDL example on the right above). This limits the maximum transferred length to 256 characters.

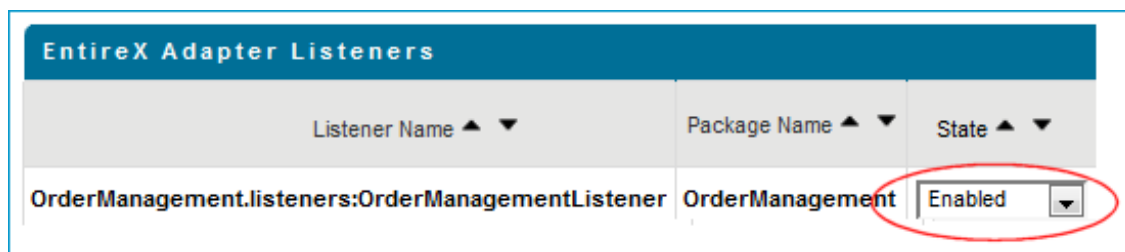
For more information on data type mapping, see *Mapping Integration Server Data Types to IDL* in the IDL Extractor for Integration Server documentation.

Testing the Extraction Results (Optional)

You can test the results of the extraction and the server back end, using the EntireX IDL Tester.

» To test the extraction results

- 1 For EntireX RPC (see [Task 3](#) in the [Introduction](#)) and an RPC Listener Connection, make sure EntireX Broker is running.
- 2 Enable the EntireX Adapter Listener.



- 3 Activate the EntireX perspective by choosing **Window > Perspective > Open Perspective > EntireX**, and from the context menu of the IDL file, choose **IDL Tester**.
- 4 This step depends on the method used to call the server (see [Task 3](#) in the [Introduction](#)).
 - For *EntireX RPC*, enter the same TCP/IP address for **Broker** that you supplied when you created the RPC Listener Connection to EntireX (localhost:1971).

- For *Direct RPC*, that is, you generated a Direct RPC Listener Connection, enter the TCP/IP address from the Integration Server (localhost:1971).
- 5 For **Server**, enter the values you specified for the server address (RPC/SRV1/CALLNAT).

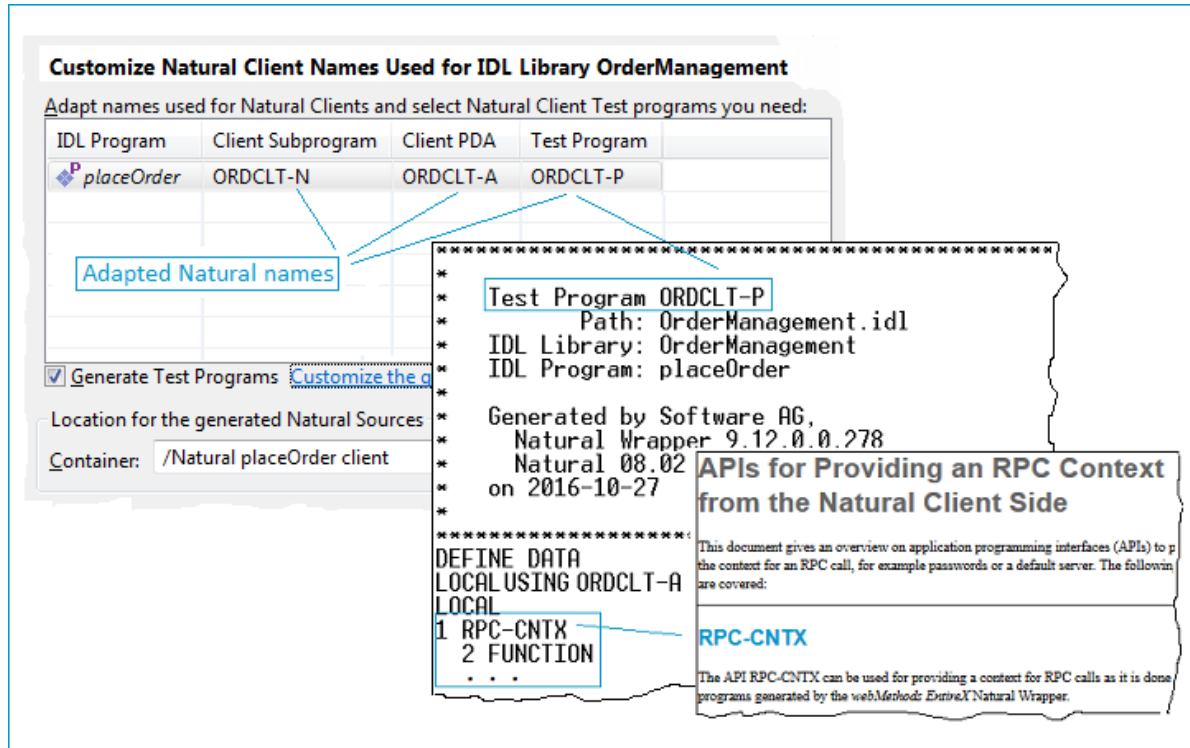
The screenshot shows a window titled 'Messages' with a list of order details. The 'Broker' field is set to 'localhost:1971' and the 'Server' field is set to 'RPC/SRV1/CALLNAT'. Both fields are circled in red. The 'Call' button is highlighted with a dashed border. The 'Messages' section displays the following data:

2 OrderItemID (N5)	45
2 OrderQuantity (N10)	18
2 OrderCustomerID (I4)	234
2 OrderItemName (A99)	tennis racquet
2 OrderUnitPrice (A20)	79.89 GBP
2 OrderTotal (A20)	1438.02 GBP
3 Name (A99)	The All England Lawn Tennis Club (Championships) Limited
3 Street (A99)	Church Road
3 City (A99)	Wimbledon London
3 ZipCode (A20)	SW19 5AE
3 CountryCode (A20)	UK
2 OrderShippingDate (A256)	Your order will be shipped on Wed, Jul 27, 2016.
2 OrderConfirmation (A256)	Your order is confirmed.
2 OrderConfirmationFlag (L)	true

Task 2: Generate Client Interface Objects and Build Client Application

➤ To generate client interface objects and build a client application

- For Natural code generation, make sure the EntireX Perspective is active: Choose **Window > Perspective > Open Perspective > EntireX**. From context menu of the IDL file, choose **Natural > Generate RPC Client**.



In the Natural Wrapper you can specify the names of the generated Natural files to match your naming conventions (see picture above). Three Natural files are generated:

- A client subprogram (extension .NSN). Also named CALLNAT. In EntireX terminology this acts as the client interface object.
- A client PDA (extension .PDA). This contains the data description of the API.
- A test program (extension .NSP). This calls the client interface object. Generation of test programs can be switched off.

Natural developers often use a single character on a fixed position in the 8-character Natural name to represent the object type of the Natural object (typically P=program; N=subprogram; A=PDA). The last character of the names from our example above uses this convention.

The test program can also be used as an example of how to code an RPC client in Natural. It uses the Natural API `RPC-CNTX`. Parameters such as host, port, server address, user ID etc. can be set for RPC communication. Especially for a quick test, this is the simplest way to set the connection parameters. See your Natural documentation for more information on API `RPC-CNTX`. You can also set connection parameters using a service directory. See *Service Directory Maintenance* in the Natural *SYSRPC Utility* documentation for more information.

Task 3: Execute the Call from Natural to Integration Server

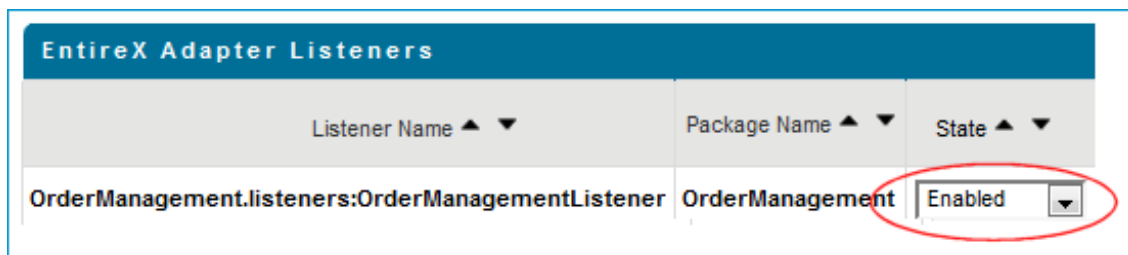
- [Generated Test Program](#)
- [IDL Tester](#)

Generated Test Program

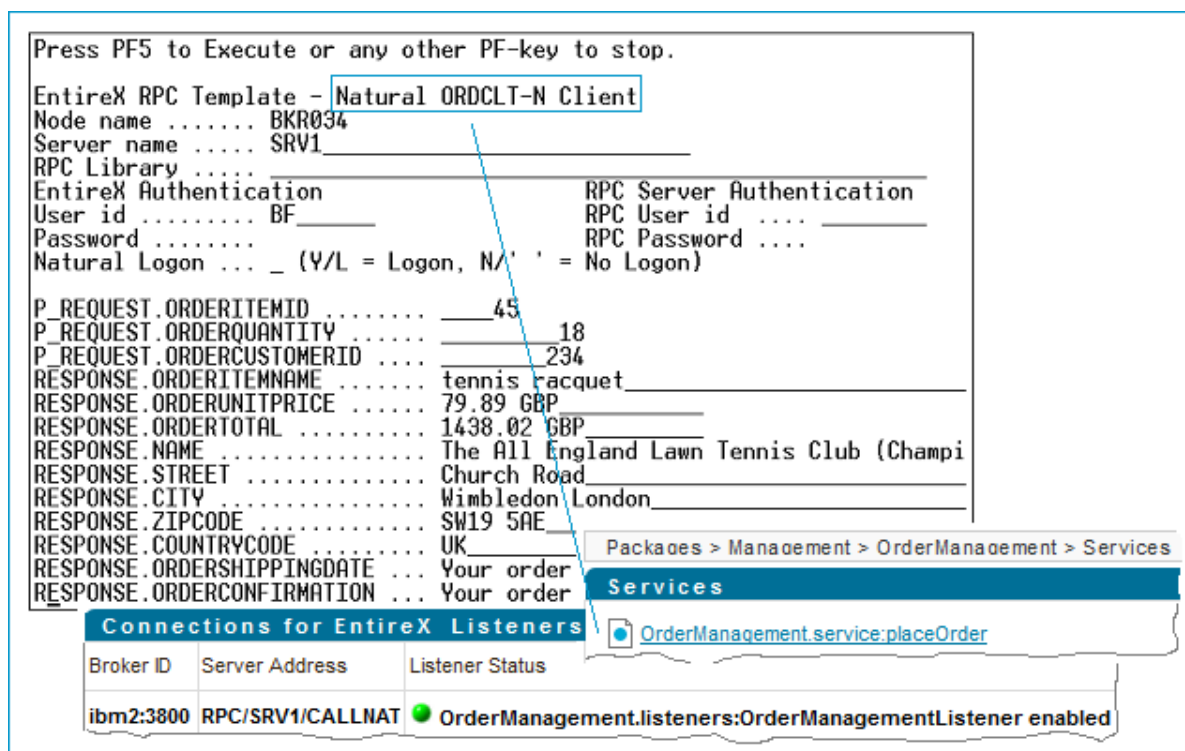
Use the generated test program to execute the call.

➤ To use the generated test program

- 1 For EntireX RPC (see [Task 3](#) in the [Introduction](#)) and an RPC Listener Connection, make sure EntireX Broker is running.
- 2 Enable the EntireX Adapter Listener.



- 3 This step depends on the method used to call the server (see [Task 3](#) in the [Introduction](#)).
 - For *EntireX RPC*, enter the same TCP/IP address for **Broker** that you supplied when you created the RPC Listener Connection to EntireX (localhost:1971).
 - For *Direct RPC*, that is, you generated a Direct RPC Listener Connection, enter the TCP/IP address from the Integration Server (localhost:1971).
- 4 Set the field `Server name` to match the middle part of the **Server** address specified under [Creating Listener Objects in Integration Server](#). Example: If you specify SRV1 in the test program, RPC/SRV1/CALLNAT will be used.

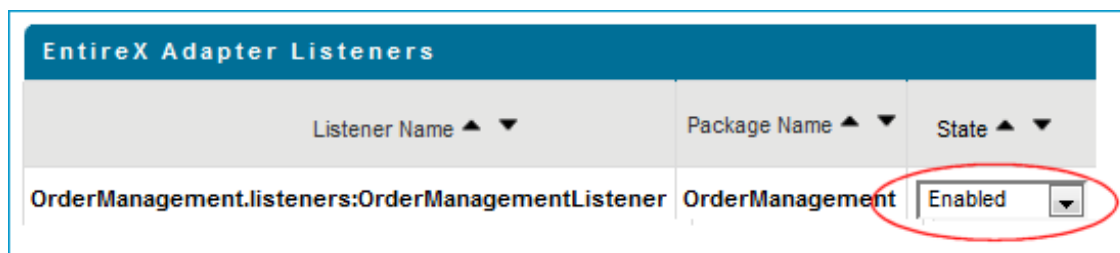


IDL Tester

Use the EntireX IDL Tester to execute the call.

> To test the extraction results

- 1 For EntireX RPC (see [Task 3](#) in the [Introduction](#)) and an RPC Listener Connection, make sure EntireX Broker is running.
- 2 Enable the EntireX Adapter Listener.



- 3 Activate the EntireX perspective by choosing **Window > Perspective > Open Perspective > EntireX**, and from the context menu of the IDL file, choose **IDL Tester**.
- 4 This step depends on the method used to call the server (see [Task 3](#) in the [Introduction](#)).

- For *EntireX RPC*, enter the same TCP/IP address for **Broker** that you supplied when you created the RPC Listener Connection to EntireX (localhost:1971).
 - For *Direct RPC*, that is, you generated a Direct RPC Listener Connection, enter the TCP/IP address from the Integration Server (localhost:1971).
- 5 For **Server**, enter the values you specified for the server address (RPC/SRV1/CALLNAT).

Broker: localhost:1971 Server: RPC/SRV1/CALLNAT

Call Ping Open Conversation Reset Exit

Messages

2 OrderItemID (N5)	45
2 OrderQuantity (N10)	18
2 OrderCustomerID (I4)	234
2 OrderItemName (A99)	tennis racquet
2 OrderUnitPrice (A20)	79.89 GBP
2 OrderTotal (A20)	1438.02 GBP
3 Name (A99)	The All England Lawn Tennis Club (Championships) Limited
3 Street (A99)	Church Road
3 City (A99)	Wimbledon London
3 ZipCode (A20)	SW19 5AE
3 CountryCode (A20)	UK
2 OrderShippingDate (A256)	Your order will be shipped on Wed, Jul 27, 2016.
2 OrderConfirmation (A256)	Your order is confirmed.
2 OrderConfirmationFlag (L)	true

