webMethods EntireX

Using EntireX RPC for CL under IBM i

Version 10.7

October 2020

WEBMETHODS

This document applies to webMethods EntireX Version 10.7 and all subsequent releases.

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

Document ID: EXX-RPC-107-20220422CL

Table of Contents

1 About this Documentation

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	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.softwareag.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 Using EntireX RPC for CL under IBM i

Note: The EntireX RPC Server for IBM i of EntireX version 7.1.1 running under IBM i (AS/400) is no longer supported. We strongly recommend using the new *RPC Server for AS/400* or the EntireX Adapter. See also *Connection Parameters for AS/400 Connections*.

Creating a Sample Server in CL

This section describes how to build a server application using the IBM i ILE language CL. The sample server will be named SENDMESS. Using the IBM i command SNGPGMMSG (send program message), it sends a message to a given IBM i user and returns a confirmation to the RPC client.

This section tells you how to

- Create the Client/Server Interface
- Create the Server
- Compile and Link the Server

Create the Client/Server Interface

Using the Designer on your PC, create a Software AG IDL file similar to the following:

```
Library 'EXAMPLE' Is
 Program 'SENDMESS' Is
  Define Data Parameter
   1 UserID           (A10)  In
   1 Message_Text     (A70)  In
   1 Confirmation     (A40)  Out
  End-Define
```

Section *Software AG IDL to CL Mapping* describes how IDL data types are mapped to CL data items.

Create the Server

The server is implemented as an ILE CL program of type *PGM.

For our IDL example SENDMESS, the implemented server looks similar to the example below:

```
       PGM        PARM(&USER &MESSTEXT &CONFIRM)
/*-------------------------------------------------------------------*/
       DCL        VAR(&USER) TYPE(*CHAR) LEN(10)     /* the user ID   */
       DCL        VAR(&MESSTEXT) TYPE(*CHAR) LEN(70) /* the text      */
       DCL        VAR(&CONFIRM) TYPE(*CHAR) LEN(40)  /* returned text */
/*-------------------------------------------------------------------*/
       CHGVAR     VAR(&CONFIRM) VALUE(' ')           /* clean it */
       SNDPGMMSG  MSG(&MESSTEXT) TOUSR(&USER) MSGTYPE(*COMP)
```

```
       MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(BAD))
       CHGVAR     VAR(&CONFIRM) +
          VALUE('Message sent to user' *BCAT &USER)
       GOTO       CMDLBL(DONE)                       /* sending was ok */
/*-------------------------------------------------------------------*/
BAD:   CHGVAR     VAR(&CONFIRM) +
          VALUE('Message sending failed')
DONE:  ENDPGM
```

Because servers are running in a multithreaded environment, your application programs must be thread-safe. This implies that all commands and subprograms accessed in your servers must allow multithreads.

Compile and Link the Server

Compile the server source using the IBM i command `CRTBNDCL` (create bound CL program).

The following example procedure demonstrates how to compile and bind an ILE CL program:

```
         PGM        /* Compile and Bind a CL Server program         */
/*-------------------------------------------------------------------*/
         DCL        VAR(&MODNAME) TYPE(*CHAR) LEN(10) VALUE(SENDMESS)
         DCL        VAR(&LIBL) TYPE(*CHAR) LEN(10) VALUE(EXAMPLE)
         DCL        VAR(&SRCF) TYPE(*CHAR) LEN(10) VALUE(QCLSRC)
         DCL        VAR(&OPTL) TYPE(*CHAR) LEN(10) VALUE(*NONE)
         DCL        VAR(&DBGV) TYPE(*CHAR) LEN(10) VALUE(*ALL)
/*-------------------------------------------------------------------*/
         MONMSG     MSGID(CPF6801) EXEC(GOTO CMDLBL(DONE))
                    /* If PF12 is pressed */

         CRTBNDCL   ??PGM(&LIBL/&MODNAME) ??SRCFILE(&LIBL/&SRCF) +
                    ??SRCMBR(&MODNAME) DFTACTGRP(*NO) +
                      ACTGRP(*CALLER) OUTPUT(*PRINT) +
                      OPTIMIZE(&OPTL) DBGVIEW(&DBGV)

         MONMSG     MSGID(LNC9001) EXEC(GOTO CMDLBL(ERRXT))
         GOTO       CMDLBL(DONE)
/*-------------------------------------------------------------------*/
 ERRXT:  SNDPGMMSG  MSG('MSG: Program Linkage Failed')
 DONE:   RETURN
         ENDPGM
```

Important: When linking/binding servers, the binding parameter `ACTGRP(*CALLER)` must be specified. This guarantees that the server application runs in the same activation group as the calling RPC Server.

Name the resulting server program like the program name in the IDL file and put it in a library whose name corresponds to the library name in the IDL file.

Example:

If a client performs a request that is based on the IDL program SENDMESS in the IDL library EXAMPLE, the ILE server program SENDMESS in the IBM i library EXAMPLE is executed. If no corresponding program can be found, the access will fail.

Note: The EntireX RPC Server for IBM i of EntireX version 7.1.1 running under IBM i (AS/400) is no longer supported. We strongly recommend using the new *RPC Server for AS/400* or the EntireX Adapter. See also *Connection Parameters for AS/400 Connections*.

Verifying the Server

To verify the server program `SENDMESS`, use the EntireX IDL Tester when the RPC Server for AS/400 and EntireX Broker are used.

To verify the server

1 Confirm that an EntireX Broker and an RPC Server for AS/400 are available in your network.

2 Make sure the IBM i host server is started on your IBM i machine.

3 Create a *Software AG IDL File* in the IDL Editor documentation using the *IDL Editor* as described under ***Create the Client/Server Interface***.

4 Start the IDL Tester.

Note: When using the EntireX Adapter, use the **Service Development** perspective of the Software AG Designer for testing.

Software AG IDL to CL Mapping

This section describes the specific mapping of Software AG IDL data types to the CL programming language. See also hints and restrictions on the Software AG IDL data types valid for all programming language bindings under *IDL Data Types* in the IDL Editor documentation.

The following topics are covered here:

- Mapping IDL Data Types to CL Data Types
- Mapping Program and Library Names
- Mapping Arrays, Groups and Structures

Mapping IDL Data Types to CL Data Types

In the table below, the following metasymbols and informal terms are used for the IDL.

- The metasymbols "[" and "]" enclose optional lexical entities.
- The informal term *number* (or in some cases *number1*.*number2*) is a sequence of numeric characters, for example 123.

Software AG IDL	Description	CL Data Type	See Notes
A*n*	Alphanumeric	TYPE(*CHAR) LEN(*n*)	
P*(n - p)*[.*p*]	Packed decimal	TYPE(*DEC) LEN(*n* [*p*])	1

See also hints and restrictions on the Software AG IDL data types valid for all programming language bindings under *IDL Data Types* in the IDL Editor documentation.

Notes:

1. *n* must be less than or equal to 15. The maximum value for *p* is 9.
   For example, the IDL definition P10.2 corresponds to TYPE(*DEC) LEN(12 2)

Other IDL data types have no appropriate equivalent in the CL language.

Mapping Program and Library Names

Do not use the special characters '#', '$', '&', '+', '-', '.', '/' and '@' within names of programs and libraries in the IDL file. These characters are not allowed within names of server programs and libraries created on IBM i.

Mapping Arrays, Groups and Structures

Arrays, Groups and Structures are not supported for the CL language.

Mapping the Direction Attributes In, Out, InOut

The IDL syntax allows you to define parameters as IN parameters, OUT parameters, or IN OUT parameters (which is the default if nothing is specified). This direction specification is reflected in the stubless call of the RPC Server as follows:

- Direction attributes do not change the call interface because parameters are always treated as “called by reference”.
- Usage of direction attributes may be useful to reduce data traffic between RPC client and RPC server.

- Parameters with the `IN` attribute are sent from the RPC client to the RPC server.
- Parameters with the `OUT` attribute are sent from the RPC server to the RPC client.
- Parameters with the `IN` and `OUT` attribute are sent from the RPC client to the RPC server and then back to the RPC client.

Note that only the direction information of the top-level fields (Level 1) is relevant. Group fields always inherit the specification from their parent. A different specification is ignored.

See the `attribute-list` under *Software AG IDL Grammar* in the IDL Editor documentation for the syntax on how to describe attributes within the Software AG IDL file and refer to `direction-attribute`.