

webMethods EntireX

EntireX RPC Server for Micro Focus COBOL

Version 10.5

October 2019

This document applies to webMethods EntireX Version 10.5 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 © 1997-2019 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-MICROFOCUSRPC-105-20220422

Table of Contents

1 About this Documentation	1
Document Conventions	2
Online Information and Support	2
Data Protection	3
2 Introduction to the RPC Server for Micro Focus	5
Worker Models	6
Inbuilt Services	7
Usage of Server Mapping Files	8
3 Administering the RPC Server for Micro Focus	9
Customizing the RPC Server	10
Configuring the RPC Server	11
Locating and Calling the Target Server	20
Using SSL/TLS with the RPC Server	21
Starting the RPC Server	23
Stopping the RPC Server	24
Pinging the RPC Server	25
Running an EntireX RPC Server as a Windows Service	26
Activating Tracing for the RPC Server	26
4 Deployment Service	27
Introduction	28
Scope	29
Enabling the Deployment Service	29
Disabling the Deployment Service	30
5 Server-side Mapping Files	31
Server-side Mapping Files in the RPC Server	32
Deploying Server-side Mapping Files to the RPC Server	33
Undeploying Server-side Mapping Files from the RPC Server	33
Change Management of Server-side Mapping Files	33
List Deployed Server-side Mapping Files	34
Check if a Server-side Mapping File Revision has been Deployed	34
Access Control: Secure Server Mapping File Deployment	34
Is There a Way to Smoothly Introduce Server-side Mapping Files?	34
6 Scenarios	35
COBOL Scenarios	36

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

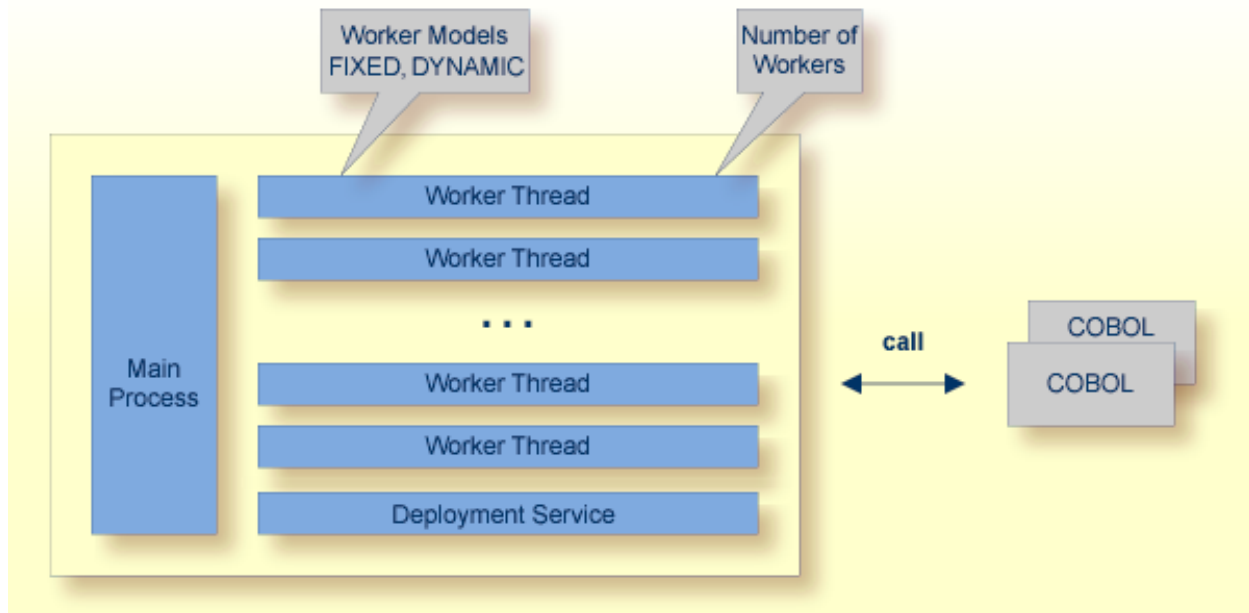
Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Introduction to the RPC Server for Micro Focus

- Worker Models 6
- Inbuilt Services 7
- Usage of Server Mapping Files 8

The EntireX RPC Server for Micro Focus COBOL allows standard RPC clients to communicate with COBOL servers written with Micro Focus COBOL. It works together with the *COBOL Wrapper* and the *IDL Extractor for COBOL*.

Worker Models



RPC requests are worked off inside the RPC server in worker threads, which are controlled by a main thread. Every RPC request occupies during its processing a worker thread. If you are using RPC conversations, each RPC conversation requires its own thread during the lifetime of the conversation. The RPC server provides two worker models:

- **FIXED**
The *fixed* model creates a fixed number of worker threads. The number of worker threads does not increase or decrease during the lifetime of an RPC server instance.
- **DYNAMIC**
The *dynamic* model creates worker threads depending on the incoming load of RPC requests.

For configuration and technical details, see parameter `workermodel` under *Administering the RPC Server for Micro Focus*.

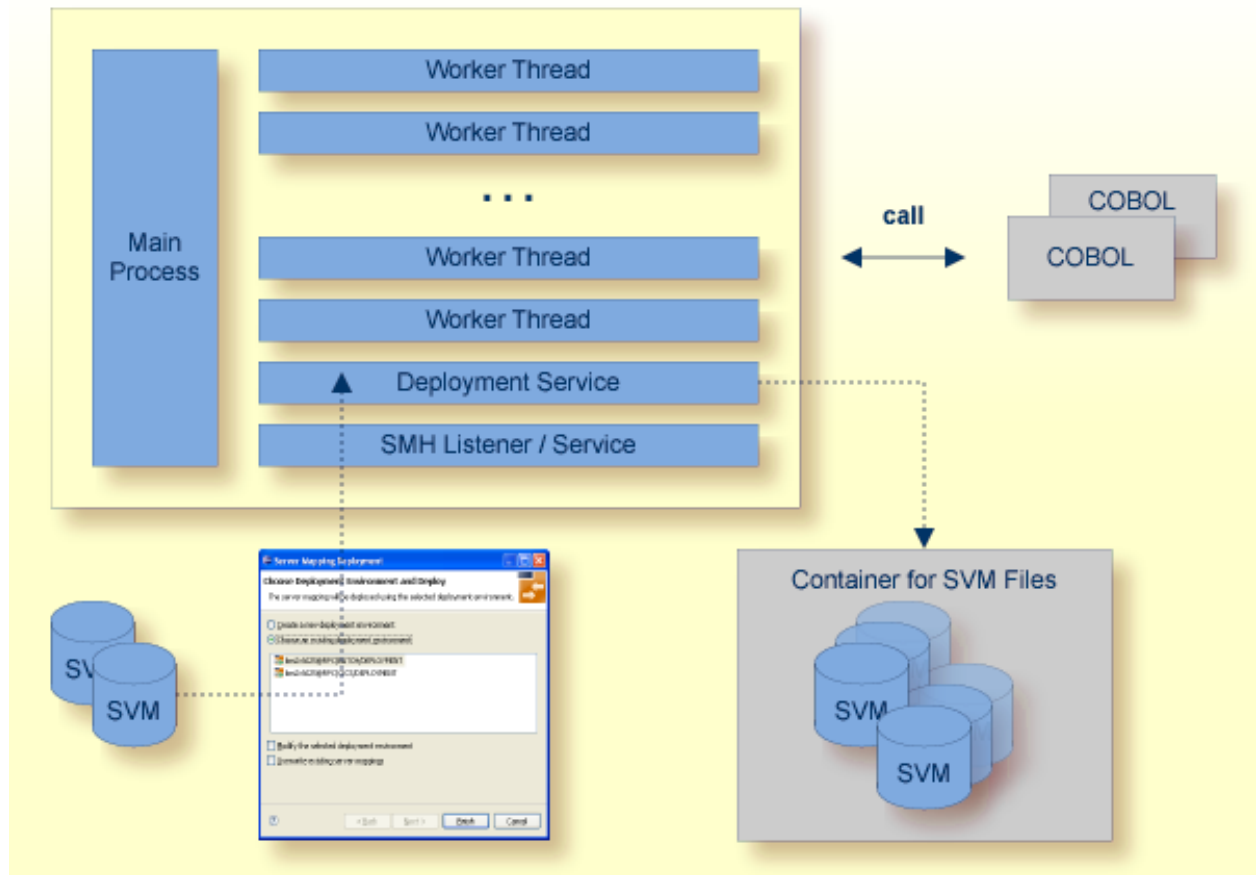
Inbuilt Services

RPC Server for Micro Focus provides the following service for ease-of-use:

- [Deployment Service](#)

Deployment Service

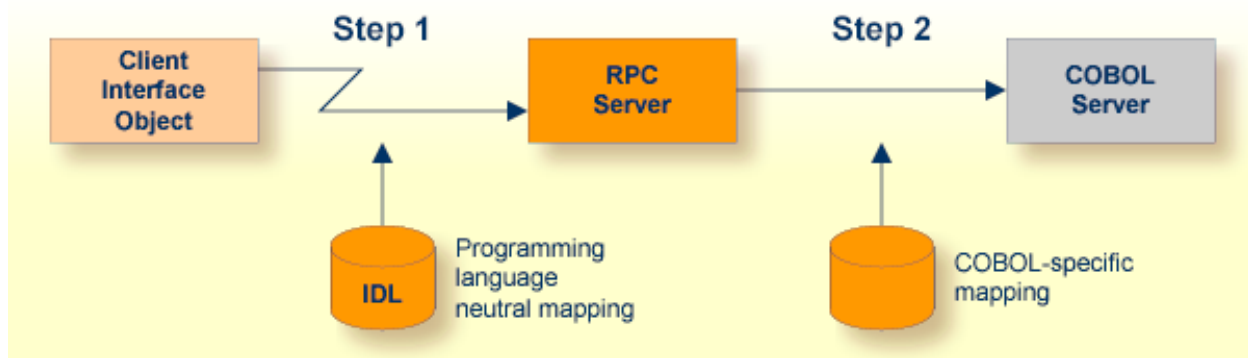
The Deployment Service allows you to deploy server-side mapping files (Designer files with extension .svm) interactively using the *Server Mapping Deployment Wizard*. On the RPC server side, the server-side mapping files are stored in a server-side mapping container (folder or directory). See [Server-side Mapping Files in the RPC Server](#) and [Deployment Service](#) for configuration information.



Usage of Server Mapping Files

There are many situations where the RPC Server for Micro Focus requires a server mapping file to correctly support special COBOL syntax such as `REDEFINES`, `SIGN LEADING` and `OCCURS DEPENDING ON` clauses, `LEVEL-88` fields, etc.

Server mapping files contain COBOL-specific mapping information that is not included in the IDL file, but is needed to successfully call the COBOL server program.



The RPC server marshals the data in a two-step process: the RPC request coming from the RPC client (Step 1) is completed with COBOL-specific mapping information taken from the server mapping file (Step 2). In this way the COBOL server can be called as expected.

The server mapping files are retrieved as a result of the IDL Extractor for COBOL extraction process and the COBOL Wrapper if a COBOL server is generated. See *When is a Server Mapping File Required?*

There are *server-side* mapping files (*Software AG Designer* files with extension `.svm`) and *client-side* mapping files (*Designer* files with extension `.cvm`). See *Server Mapping Files for COBOL* and *How to Set the Type of Server Mapping Files*.

If you are using server-side mapping files, you need to customize the server-side mapping container with parameter `svm`. See *Configuring the RPC Server*.

3 Administering the RPC Server for Micro Focus

- Customizing the RPC Server 10
- Configuring the RPC Server 11
- Locating and Calling the Target Server 20
- Using SSL/TLS with the RPC Server 21
- Starting the RPC Server 23
- Stopping the RPC Server 24
- Pinging the RPC Server 25
- Running an EntireX RPC Server as a Windows Service 26
- Activating Tracing for the RPC Server 26

The EntireX RPC Server for Micro Focus COBOL allows standard RPC clients to communicate with COBOL servers written with Micro Focus COBOL. It works together with the *COBOL Wrapper* and the *IDL Extractor for COBOL*.

Customizing the RPC Server

The following elements are used for setting up the RPC Server for Micro Focus:

- [Micro Focus COBOL Runtime](#)
- [Configuration File](#)
- [Start Script](#)

Micro Focus COBOL Runtime

The COBOL runtime, for example *Micro Focus Server*, has to be installed according to the Micro Focus documentation. It is not delivered with this package. Provide the location of the COBOL runtime in the *Start Script*.

If a COBOL runtime is not provided, the RPC Server for Micro Focus cannot be started and an error message is given.

Configuration File

The name of the delivered example configuration file is *microfocussserver.cfg* provided in the *config* folder. The configuration file contains the configuration for the RPC Server for Micro Focus. The following settings are important:

- connection information such as broker ID, server address (class, name, service)
- location and usage of server-side mapping container; see [Usage of Server Mapping Files](#).
- scalability parameters
- trace settings
- etc.

For more information see [Configuring the RPC Server](#).

Start Script

The start script for the RPC Server for Micro Focus is called *microfocusserver.bsh* (UNIX) or *microfocusserver.bat* (Windows) and is provided in the *bin* folder of the installation directory. You may customize this file. The start script contains the following:

- location of the Micro Focus COBOL runtime
- paths to the called COBOL server; see [Configuration Approaches](#)
- the configuration file used; see [Configuration File](#)
- etc.

Configuring the RPC Server

The following rules apply:

- In the configuration file:
 - Comments must be on a separate line.
 - Comment lines can begin with '*', '/' and ';'.
 - Empty lines are ignored.
 - Headings in square brackets [<topic>] are ignored.
 - Keywords are not case-sensitive.
- Underscored letters in a parameter indicate the minimum number of letters that can be used for an abbreviated command.

For example, in `brokerid=localhost`, `brok` is the minimum number of letters that can be used as an abbreviation, that is, the commands/parameters `broker=localhost` and `brok=localhost` are equivalents.

Parameter	Default	Values	Req/Opt
<code>brokerid</code>	localhost	Broker ID used by the server. See <i>Using the Broker ID in Applications</i> . Example: <code>brokerid=myhost.com:1971</code>	R
<code>class</code>	RPC	Server class part of the server address used by the server. The server address must be defined as a service in the broker attribute file (see <i>Service-specific Attributes</i>). Case-sensitive, up to 32 characters. Corresponds to CLASS.	R

Parameter	Default	Values	Req/ Opt
		Example: <code>class=MyRPC</code>	
<code>codepage</code>		<p>The codepage tells the broker the encoding of the data. The application must ensure the encoding of the data matches the codepage. The RPC server itself does not convert your application data. The application's data is shipped and received as given. Often, the codepage must also match the encoding used in the RPC server environment for file and terminal IO, otherwise unpredictable results may occur.</p> <p>Under the Windows operating system:</p> <ul style="list-style-type: none"> ■ By default, the Windows ANSI codepage configured for your system is automatically transferred to tell the broker how the data is encoded. ■ If you want to adapt the Windows ANSI codepage, see the Regional Settings in the Windows Control Panel and your Windows documentation. ■ If you want to encode the data different to your Windows ANSI codepage, convert the data in the application and provide the codepage name here. During receive, decode the data accordingly. <p>Under the UNIX operating system:</p> <ul style="list-style-type: none"> ■ By default, no codepage is transferred to the broker. ■ It is assumed the broker's locale string defaults match. See <i>Locale String Mapping</i>. If they do not match, provide the codepage here. Example: <pre style="background-color: #f0f0f0; padding: 5px;">codepage=iso-8859-1</pre> <p>Enable character conversion in the broker by setting the service-specific attribute <code>CONVERSION</code> to "SAGTRPC". See also <i>Configuring ICU Conversion under Configuring Broker for Internationalization</i> in the platform-specific Administration documentation. More information can be found under <i>Internationalization with EntireX</i>.</p>	R/O

Parameter	Default	Values	Req/Opt
<u>compresslevel</u>	N	<p>Enforce compression when data is transferred between broker and server. See <i>Data Compression in EntireX Broker</i>.</p> <p><code>compresslevel= 0 1 2 3 4 5 6 7 8 9 Y N</code></p> <p>0-9 0=no compression 9=max. compression</p> <p><u>N</u> No compression. <u>Y</u> Compression level 6.</p> <p>Example: <code>compresslevel=6</code></p>	O
<u>deployment</u>	NO	<p>Activates the deployment service, see Deployment Service. Required to use the Server Mapping Deployment Wizard. See <i>Server Mapping Deployment Wizard</i> in the Designer documentation.</p> <p><u>YES</u> Activates the deployment service. The RPC server registers the deployment service in the broker.</p> <p><u>NO</u> The deployment service is deactivated. The RPC server does not register the deployment service in the broker.</p> <p>Example: <code>deployment=yes</code></p>	O
<u>logon</u>	YES	<p>Execute broker functions LOGON/LOGOFF in worker threads. Must match the setting of the broker attribute AUTOLOGON. Reliable RPC requires logon set to YES. See <i>Reliable RPC</i>.</p> <p><u>NO</u> No logon/logoff functions are executed. <u>YES</u> Logon/logoff functions are executed.</p> <p>Example: <code>logon=no</code></p>	O
<u>marshalling</u>	COBOL	<p>The RPC Server for Micro Focus supports COBOL. See also Locating and Calling the Target Server. <code>Marshalling=(LANGUAGE=COBOL, flavor=MF)</code> must be provided. Do not change these settings. The COBOL servers are called directly without a server interface object. So-called server mapping files are</p>	O

Parameter	Default	Values	Req/Opt
		used to call the COBOL server correctly if one is available. See <i>Usage of Server Mapping Files</i> .	
<code>password</code>	no default	The password for secured access to the broker. If possible (write access) the password is encrypted and written to parameter <code>password.e</code> . The parameter <code>password</code> is removed. To change the password, add the parameter <code>password</code> with the new password as value. Example: <code>password=MyPwd</code>	O
<code>restartcycles</code>	15	Number of restart attempts if the broker is not available. This can be used to keep the RPC Server for Micro Focus running while the broker is down for a short time. A restart cycle will be repeated every 60 seconds. Note: Internally, the server waits in periods of 10 seconds (performing six times more cycles), which you can see in the server output. When the number of specified cycles is reached and a connection to the broker is not possible, the RPC Server for Micro Focus stops. Example: <code>restartcycles=30</code> The server waits up to 30 minutes (30*6*10 seconds) before it terminates due to a missing broker connection.	O
<code>servername</code>	SRV1	Server name part of the server address used by the server. The server address must be defined as a service in the broker attribute file. See <i>Service-specific Attributes</i> . Case-sensitive, up to 32 characters. Corresponds to <code>SERVER</code> of the broker attribute file. Example: <code>servername=mySrv</code>	R
<code>service</code>	CALLNAT	Service part of the server address used by the server. The server address must be defined as a service in the broker attribute file. See <i>Service-specific Attributes</i> . Case-sensitive, up to 32 characters. Corresponds to <code>SERVICE</code> attribute of the broker attribute file. Example: <code>service=MYSERVICE</code>	R

Parameter	Default	Values	Req/ Opt
<code>ssl_file</code>	no default	Set the SSL parameters. See Using SSL/TLS with the RPC Server for examples and more information.	O
<code>svm</code>		Usage and anchor of the server-side mapping container (directory or folder). See Server-side Mapping Files in the RPC Server . The RPC server needs write access to the server-side mapping container. There are also client-side mapping files that do not require configuration here. See Server Mapping Files for COBOL SVM=(PATH= <i>path</i>) <i>path</i> The path to the anchor of the server-side mapping container. Example for UNIX: SVM=(PATH=../config/svm) Example for Windows: SVM=(PATH=..\config\svm) See also Usage of Server Mapping Files .	O
<code>timeout</code>	60	Timeout in seconds, used by the server to wait for broker requests. See broker ACI control block field WAIT for more information. Also influences restartcycles and worker model DYNAMIC . Example: timeout=300	O
<code>tracedestination</code>	ERXTrace. <i>nnn</i> .log	The name of the destination file for trace output. By default the main trace file name is ERXTrace. <i>nnn</i> .log, where <i>nnn</i> can be in the range from 001 to 005. See also Activating Tracing for the RPC Server . ■ Under UNIX, the trace file is located in the current working directory. ■ Under Windows, the trace file is located in a subfolder of the windows folder <i>My Documents</i> . If the default is not used and a <code>tracedestination</code> is specified, you can use the following variables depending on the operating system: %...% Windows Environment variable.	O

Parameter	Default	Values	Req/ Opt
		<p>\$(...)</p> <p>UNIX Environment variable.</p> <p>@PID UNIX, Win Process ID.</p> <p>@TID UNIX, Win Thread ID.</p> <p>@RANGE[<i>n, m</i>] UNIX, Win <i>m</i> must be greater than <i>n</i>, range is from 0 - 999</p> <p>@CSIDL_PERSONAL Windows The user's home directory. The variable will be resolved by Windows shell functions.</p> <p>@CSIDL_APPDATA Windows The <i>Application Data Directory</i>. The variable will be resolved by Windows shell functions.</p> <p>@CSIDL_LOCAL_APPDATA Windows The <i>Local Application Data Directory</i>. The variable will be resolved by Windows shell functions.</p> <p>See also Activating Tracing for the RPC Server.</p> <p>Example: tracedestination=ERXTrace.log</p>	
tracelevel	None	<p>Trace level for the server. See also Activating Tracing for the RPC Server.</p> <p>tracelevel = <u>None</u> Standard Advanced ↔ Support</p> <p>None No trace output. Standard For minimal trace output. Advanced For detailed trace output.</p>	O

Parameter	Default	Values	Req/ Opt
		<p>Support This trace level is for support diagnostics and should only be switched on when requested by Software AG support.</p> <p>Example: tracelevel=standard</p>	
<u>traceoption</u>	None	<p>Additional trace option if trace is active. See also Activating Tracing for the RPC Server.</p> <p>None No additional trace options.</p> <p>STUBLOG If tracelevel is Advanced or Support, the trace additionally activates the broker stub log.</p> <p>NOTRUNC Normally if a data buffer larger than 8 KB is traced, the buffer trace is truncated. Set this option to write the full amount of data without truncation.</p> <p>Note: This can increase the amount of trace output data dramatically if you transfer large data buffers.</p> <p>Example: traceoption=(STUBLOG,NOTRUNC)</p>	O
<u>userid</u>	ERX-SRV	<p>The user ID for access to the broker. The default ERX-SRV will be used if this parameter is omitted or specified without a value: "userid=".</p> <p>Example: userid=MyUid</p>	O
<u>workermodel</u>	SCALE,1,3,slowshrink	<p>The RPC Server for Micro Focus can be configured to</p> <ul style="list-style-type: none"> ■ use a DYNAMIC worker model, which adjusts the number of worker threads to the current number of client requests: 	O

Parameter	Default	Values	Req/ Opt								
		<pre data-bbox="743 281 1222 443">workermodel=(SCALE,from,thru [,slowshrink ← fastshrink] [,noisolation ← isolation])</pre> <p data-bbox="716 485 1211 512">■ use a FIXED number of worker threads:</p> <pre data-bbox="743 558 1190 653">workermodel=(FIXED,number [,noisolation ← isolation])</pre> <table border="1" data-bbox="711 716 1325 1896"> <tr> <td data-bbox="711 716 889 831">FIXED</td> <td data-bbox="889 716 1325 831">A fixed <i>number</i> of worker threads is used by the RPC Server for Micro Focus.</td> </tr> <tr> <td data-bbox="711 831 889 1356">SCALE</td> <td data-bbox="889 831 1325 1356">The number of worker threads is adjusted to the current number of client requests. With the <i>from</i> value, the minimum number of active worker threads can be set. This allows you to define a certain number of threads - not used by the currently executing RPC request - to wait for new RPC client requests to process. In this way the RPC server is ready to handle many RPC client requests arriving at the same time. The <i>thru</i> value restricts the maximum number of all worker threads concurrently.</td> </tr> <tr> <td data-bbox="889 1356 1084 1713">slowshrink</td> <td data-bbox="1084 1356 1325 1713">Default. The RPC server stops all worker threads not used in the time specified by the timeout parameter, except for the number of workers specified as minimum value.</td> </tr> <tr> <td data-bbox="889 1713 1084 1896">fastshrink</td> <td data-bbox="1084 1713 1325 1896">The RPC server stops worker threads immediately as soon as it has</td> </tr> </table>	FIXED	A fixed <i>number</i> of worker threads is used by the RPC Server for Micro Focus.	SCALE	The number of worker threads is adjusted to the current number of client requests. With the <i>from</i> value, the minimum number of active worker threads can be set. This allows you to define a certain number of threads - not used by the currently executing RPC request - to wait for new RPC client requests to process. In this way the RPC server is ready to handle many RPC client requests arriving at the same time. The <i>thru</i> value restricts the maximum number of all worker threads concurrently.	slowshrink	Default. The RPC server stops all worker threads not used in the time specified by the timeout parameter, except for the number of workers specified as minimum value.	fastshrink	The RPC server stops worker threads immediately as soon as it has	
FIXED	A fixed <i>number</i> of worker threads is used by the RPC Server for Micro Focus.										
SCALE	The number of worker threads is adjusted to the current number of client requests. With the <i>from</i> value, the minimum number of active worker threads can be set. This allows you to define a certain number of threads - not used by the currently executing RPC request - to wait for new RPC client requests to process. In this way the RPC server is ready to handle many RPC client requests arriving at the same time. The <i>thru</i> value restricts the maximum number of all worker threads concurrently.										
slowshrink	Default. The RPC server stops all worker threads not used in the time specified by the timeout parameter, except for the number of workers specified as minimum value.										
fastshrink	The RPC server stops worker threads immediately as soon as it has										

Parameter	Default	Values		Req/ Opt
			finished its conversation, except for the number of workers specified as minimum value.	
		noisolation	Calls to the COBOL server are executed within the RPC Server for Micro Focus. If the COBOL server causes a COBOL runtime error, the RPC Server for Micro Focus stops.	
		isolation	Default. Calls to the COBOL server are executed in separate processes. If the COBOL server causes a COBOL runtime error, the RPC Server for Micro Focus does not stop and continues.	
		Example: workermodel=(SCALE,2,5)		

Locating and Calling the Target Server

Introduction

The RPC Server for Micro Focus is able to call standard libraries (Windows DLLs or UNIX shared objects/libraries); Micro Focus proprietary formats such as intermediate code (*.int); generated code (*.gnt); and intermediate or generated code packaged in libraries (*.lbr). See the following table:

Executable Format	File Extension	File Name	Entry Point	Notes	Configuration
Operating system standard library with multiple server	.so sl (UNIX) or .dll (Windows)	IDL library	IDL program	1,2	1
Operating system standard library with single server	.so sl (UNIX) or .dll (Windows)	IDL program	IDL program	1,3,4	2
Micro Focus proprietary intermediate code	.int	IDL program		4	2
Micro Focus proprietary generated code	.gnt	IDL program		4	2
Micro Focus proprietary library with multiple server	.lbr	IDL library	IDL program	2,5	2
Micro Focus proprietary library with single server	.lbr	IDL program	IDL program	3,4,5	2

Notes

1. This type of library is a standard library (UNIX shared library or Windows DLL).
2. This type of library may contain multiple COBOL servers. The IDL library name is used to form the operating system file name. The COBOL server names (entry points) are taken as follows:
 - if the COBOL Wrapper is used, by default from the IDL program names. The IDL program name can be different if it is renamed during the wrapping process, see *Customize Automatically Generated Server Names*
 - if the IDL Extractor for COBOL is used, from the COBOL program IDs. The IDL program name can be different if it is renamed during the extraction process in the *COBOL Mapping Editor*

If the IDL program name is different, a server mapping is required, See [Usage of Server Mapping Files](#).

3. This type of library must contain one COBOL server only.
4. The IDL library name is not used. The COBOL server name (operating system file name and its entry point) are taken as follows:

- if the COBOL Wrapper is used, by default from the IDL program name. The IDL program name can be different if it is renamed during the wrapping process, see *Customize Automatically Generated Server Names*
- if the IDL Extractor for COBOL is used, from the COBOL program ID. The IDL program name can be different if it is renamed during the extraction process in the *COBOL Mapping Editor*

If the IDL program name is different, a server mapping is required, See [Usage of Server Mapping Files](#).

5. Intermediate (*.int) or generated (*.gnt) code must be packaged in the library.

Configuration Approaches

There are two approaches to access the COBOL server during runtime, which depend on the executable format (see table above):

1. The operating system's standard call mechanism is used to call libraries. Make sure your server(s) are accessible, for example:
 - under UNIX with the `LD_LIBRARY_PATH` environment variable
 - under Windows with the `PATH` environment variable
2. The Micro Focus environment variable `COBPATH` must be set before starting the RPC server. It lists all paths where a search for COBOL servers is to be performed. See the Micro Focus documentation for more information.

For both approaches, the start script of the RPC Server for Micro Focus is an appropriate place to set the environment variables. See [Start Script](#).

See also [Scenario I: Calling an Existing COBOL Server](#) or [Scenario II: Writing a New COBOL Server](#).

Using SSL/TLS with the RPC Server

RPC servers can use Secure Sockets Layer/Transport Layer Security (SSL/TLS) as the transport medium. The term “SSL” in this section refers to both SSL and TLS. RPC-based servers are always SSL clients. The SSL server can be either the EntireX Broker, Broker SSL Agent, or Direct RPC in webMethods Integration Server (IS inbound). For an introduction see *SSL/TLS and Certificates with EntireX* in the Platform-independent Administration documentation.

➤ To use SSL

- 1 To operate with SSL, certificates need to be provided and maintained. Depending on the platform, Software AG provides default certificates, but we strongly recommend that you

create your own. See *SSL/TLS Sample Certificates Delivered with EntireX* in the EntireX Security documentation.

2 Specify the Broker ID, using one of the following styles:

- *URL Style*, for example:

```
ssl://localhost:2010
```

- *Transport-method Style*, for example:

```
ETB024:1609:SSL
```

If no port number is specified, port 1958 is used as default.

3 Specify SSL parameters, using one of the methods below:

- **As part of the Broker ID**

The simplest way to specify short SSL parameter is to add them to the Broker ID.

Example with URL-style Broker ID:

```
ssl://localhost:2010?VERIFY_SERVER=N&TRUST_STORE=c:\\certs\\CaCert.pem
```

Example with transport-method-style Broker ID:

```
ETB024:1609:SSL?VERIFY_SERVER=N&TRUST_STORE=c:\\certs\\CaCert.pem
```

- **In the SSL file**

Complex SSL parameters can be specified in a so-called SSL file, a text file containing the parameters.

1. Define the SSL file with the SSL parameters, for example file *mySSLParms.txt* with the following contents:

```
VERIFY_SERVER=N  
TRUST_STORE=c:\\certs\\CaCert.pem
```

2. Define the SSL file in the configuration file of the RPC Server for Micro Focus. See parameter `ssl_file` under *Configuring the RPC Server*. Example:

```
brokerid=ssl://localhost:2010
.
.
ssl_file=C:\mySSLdirectory\mySSLParms.txt
```

If the SSL client checks the validity of the SSL server only, this is known as *one-way SSL*. The mandatory `trust_store` parameter specifies the file name of a keystore that must contain the list of trusted certificate authorities for the certificate of the SSL server. By default a check is made that the certificate of the SSL server is issued for the hostname specified in the Broker ID. The common name of the subject entry in the server's certificate is checked against the hostname. If they do not match, the connection will be refused. You can disable this check with SSL parameter `verify_server=no`.

If the SSL server additionally checks the identity of the SSL client, this is known as *two-way SSL*. In this case the SSL server requests a client certificate (the parameter `verify_client=yes` is defined in the configuration of the SSL server). Two additional SSL parameters must be specified on the SSL client side: `key_store` and `key_passwd`. This keystore must contain the private key of the SSL client. The password that protects the private key is specified with `key_passwd`.

The ampersand (&) character cannot appear in the password.

SSL parameters are separated by ampersand (&). See also *SSL/TLS Parameters for SSL Clients*.

- 4 Make sure the SSL server to which the RPC Server for Micro Focus connects is prepared for SSL connections as well. The SSL server can be EntireX Broker, Broker SSL Agent, or Direct RPC in webMethods Integration Server (IS inbound). See:
 - *Running Broker with SSL/TLS Transport* in the platform-specific Administration documentation
 - Broker SSL Agent in the UNIX and Windows Administration documentation
 - *Support for SSL/TLS* in the EntireX Adapter documentation (for Direct RPC)

Starting the RPC Server

Before starting, make sure all your (customer-written) COBOL servers are accessible through the standard Windows DLL or UNIX shared library/object load mechanism. See also [Locating and Calling the Target Server](#).

» To start the RPC Server for Micro Focus

- Use the [Start Script](#).

Or:

Use the following format:

```
rpcserver CFG=name [-option] [brokerid] [class] [servername] [service]
```

Here are some sample options. See [Configuring the RPC Server](#) for full list.

<code>-serverlog <i>file</i></code>	Defines an alternative log file. Under Windows, this is typically used by Windows Services. See Running an EntireX RPC Server as a Windows Service .
<code>-s[ilent]</code>	Run the RPC server in silent mode, that is, no terminal input will be required (for example to acknowledge error messages). The batch scripts will terminate automatically. Under UNIX, this is recommended when running in background mode.
<code>-TraceDestination <i>file</i></code>	Set the trace destination parameter.
<code>-TraceLevel None Standard Advanced</code>	Set the trace level parameter.



Note: The server input arguments are resolved from left to right. Parameters defined in the configuration file may be overridden by parameters applied on the command line and vice versa. See [Configuring the RPC Server](#) for full list of options.

Or:

Under Windows you can use the RPC Server for Micro Focus as a Windows Service. See [Running an EntireX RPC Server as a Windows Service](#).

Stopping the RPC Server

» To stop the RPC Server for Micro Focus

- Use the command `stopService`. See [Stop Running Services](#) in Command Central's Command-line Interface.

Or:

Stop the service using Command Central's Graphical User Interface. See [Stopping a Service](#).

Or:

Use the command-line utility `etbcmd`. See `etbcmd` under [Broker Command-line Utilities](#) in the platform-specific Administration documentation.

Or:

Use CTRL-C in the session where you started the RPC server instance.

Or:

Under UNIX, enter command `kill -process-id`.

See also *Component Return Codes in EntireX*.

Pinging the RPC Server

➤ To ping the RPC Server for Micro Focus

- Enter the following command:

```
java -classpath "$EXXDIR/classes/entirex.jar" ↵  
com.softwareag.entirex.rpcping.RPCServerPing -p <admin_port>
```

where *admin_port* is the number of the administration port.

The `ping` command returns "0" if the server is reachable, and "1" if the server cannot be accessed.

Running an EntireX RPC Server as a Windows Service

For general information see *Running an EntireX RPC Server as a Windows Service*.

➤ To run the RPC Server for Micro Focus as a Windows Service

- 1 Customize the *Start Script* according to your system installation.



Note: The script file must pass external parameters to the RPC server and use the option `-silent:`

```
rpcserver CFG=..\config\microfocusserver.cfg -s %*
```

See also *Starting the RPC Server*.

- 2 Test your RPC server to see whether it will start if you run your script file.
- 3 Use the *EntireX RPC Service Tool* and install the `RPCService` with some meaningful extension, for example `MyServer`. If your *Start Script* is *microfocusserver.bat*, the command will be

```
RPCService -install -ext MyServer ↵  
-script install_path\EntireX\bin\microfocusserver.bat
```

The log file will be called `RPCservice_MyServer.log`.

- 4 In **Windows Services** menu (**Control Panel** > **Administrative Tools** > **Services**) select the **service: Software AG EntireX RPC Service [MyServer]** and change the property **Startup Type** from "Manual" to "Automatic".

Activating Tracing for the RPC Server

➤ To switch on tracing for the RPC Server for Micro Focus

- 1 Set the parameters `tracelevel`, `traceoption` and `tracedestination`. See *Configuring the RPC Server*.
- 2 Start the RPC Server for Micro Focus. See *Starting the RPC Server*.
- 3 To evaluate the return codes, see *Component Return Codes in EntireX*.

➤ To switch off tracing

- Set the `tracelevel` parameter to `None`.

4 Deployment Service

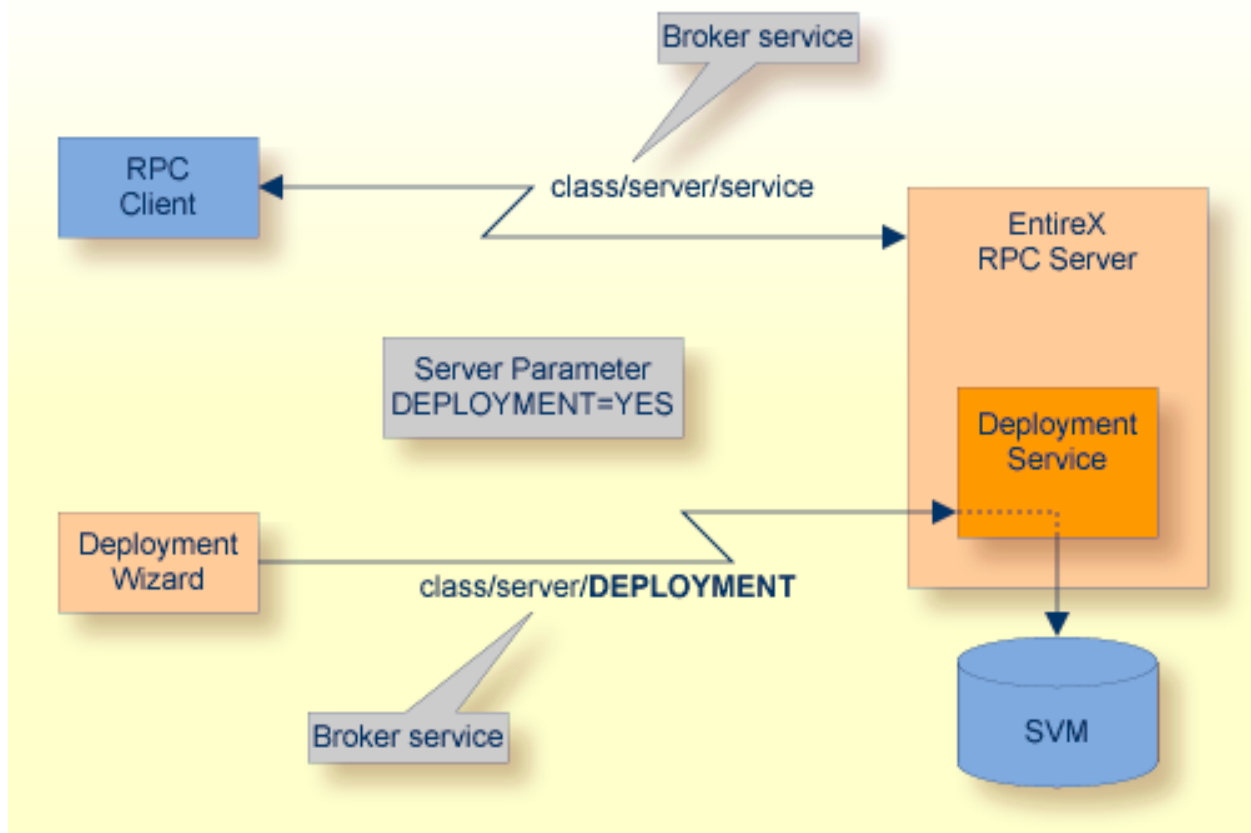
- Introduction 28
- Scope 29
- Enabling the Deployment Service 29
- Disabling the Deployment Service 30

Introduction

The deployment service is the (server-side) counterpart to the deployment wizard; see *Server Mapping Deployment Wizard*. It is a built-in service of the EntireX RPC server, which can be enabled/disabled by EntireX RPC server configuration settings.

Usage can be restricted to certain users or group of users, using EntireX Security; see *Authorization of Client and Server* in the EntireX Security documentation.

You need to configure the deployment service only when server-side mapping files are used. There are also client-side server mapping files that do not need configuration here; see *Server Mapping Files for COBOL* in the Designer documentation.



Scope

The deployment service is used in conjunction with the

- IDL Extractor for COBOL to deploy server-side mapping files with the deployment wizard;
- COBOL Wrapper for RPC server generation to deploy server-side mapping files with the deployment wizard.

See also [Deploying Server-side Mapping Files to the RPC Server](#).

The deployment service uses the same class and server names as defined for the EntireX RPC server, and `DEPLOYMENT` as the service name, resulting in `class/server/DEPLOYMENT` as the broker service. Please note `DEPLOYMENT` is a service name reserved by Software AG. See broker attribute `SERVICE`.

Enabling the Deployment Service

» To enable the deployment service

- 1 For an RPC Server for Micro Focus, configure the server mapping file subparameter `path` of parameter `svm` to point to a directory with write access. See [Configuring the RPC Server](#).
- 2 Set the RPC server parameter `deployment=yes`. See [deployment](#) under [Configuring the RPC Server](#).
- 3 Define in the broker attribute file, under the RPC service, an additional broker service with `DEPLOYMENT` as the service name and values for class and server identical to those used for the RPC service. For example, if your RPC service is named

```
CLASS = RPC    SERVER = SRV1    SERVICE = CALLNAT
```

the deployment service requires the following additional service definition in the broker attribute file:

```
CLASS = RPC    SERVER = SRV1    SERVICE = DEPLOYMENT
```

- 4 Optional. If you need to restrict the use of the deployment service to a selected group of users, use EntireX Security and define security rules for the `class/server/DEPLOYMENT` broker service. The service name `DEPLOYMENT` is a constant.
 - For a z/OS broker, see [Resource Profiles in EntireX Security](#).
 - For a UNIX or Windows broker, see [Authorization Rules](#).

- Not applicable to a BS2000 or z/VSE broker.

Disabling the Deployment Service

> To disable the deployment service

- Set the RPC Server for Micro Focus parameter `deployment=no`. See [deployment](#) under *Configuring the RPC Server*.

The RPC Server for Micro Focus will not register the deployment service in the broker.

5 Server-side Mapping Files

- Server-side Mapping Files in the RPC Server 32
- Deploying Server-side Mapping Files to the RPC Server 33
- Undeploying Server-side Mapping Files from the RPC Server 33
- Change Management of Server-side Mapping Files 33
- List Deployed Server-side Mapping Files 34
- Check if a Server-side Mapping File Revision has been Deployed 34
- Access Control: Secure Server Mapping File Deployment 34
- Is There a Way to Smoothly Introduce Server-side Mapping Files? 34

Server mapping enables the RPC server to correctly support special COBOL syntax such as `REDEFINES`, `SIGN LEADING` and `OCCURS DEPENDING ON` clauses, `LEVEL-88` fields, etc. If one of these elements is used, the IDL Extractor for COBOL automatically extracts a server mapping file in addition to the IDL file (interface definition language). Also, the COBOL Wrapper may generate a server mapping file for RPC server generation. The server mapping is used at runtime to marshal and unmarshal the RPC data stream. There are client-side mapping files (Designer files with extension `.cvm`) and server-side mapping files (Designer files with extension `.svm`). If you have not used server-side mapping, we recommend you use client-side mapping. See *Server Mapping Files for COBOL* in the Designer documentation.

See also *Source Control of Server Mapping Files* | *Comparing Server Mapping Files* | *When is a Server Mapping File Required?* | *Migrating Server Mapping Files* in the Designer documentation.

Server-side Mapping Files in the RPC Server

For RPC Server for Micro Focuss under UNIX or Windows, server-side mapping corresponds to lines of Designer files with extension `.svm`. See *Server Mapping Files for COBOL*. The server-side mapping is stored as directories (folders) and operating system files. For each IDL library, a directory is created by the deployment service during deployment and each server mapping related to an IDL program is stored as an operating system file within this directory containing the server mapping. The anchor of the server-side mapping container (directory or folder) is configured by the server mapping file subparameter "path" of parameter "svm". See *Configuring the RPC Server*. For example, deploying the file `example.svm` from the EntireX directory `examples/RPC/basic/example` results in folder `EXAMPLE` and operating system files for the IDL programs `CALC` and `SQUARE`:

```
../EXAMPLE  
/CALC.svm  
/SQUARE.svm
```

If *one* server requires a server-side mapping file, you need to provide this to the RPC server:

- Development environments: to deploy new server-side mapping files, see [Deploying Server-side Mapping Files to the RPC Server](#).
- Production environments: provide a server-side mapping container (directory or folder) containing all required server-side mapping files to the RPC server. See configuration parameter `svm`.

If *no* server requires server-side mapping, you can execute the RPC server without a server-side mapping container (directory or folder).

- Development environments: you can disable the deployment service. See [Disabling the Deployment Service](#).
- Production environments: there is no need to provide a server-side mapping container (directory or folder) to the RPC server. See configuration parameter `svm`.

Deploying Server-side Mapping Files to the RPC Server

Deploy a server-side mapping file (Designer file with extension .svm) with the Server Mapping Deployment Wizard. See *Server Mapping Files for COBOL* in the Designer documentation.

➤ **To deploy a server-side mapping file with the Server Mapping Deployment Wizard**

- 1 Make sure the RPC server is active and that the Deployment Service of the RPC server is properly configured. See [Deployment Service](#).
- 2 From the context menu of your IDL file, choose **COBOL > Deploy/Synchronize Server Mapping** COBOL > Deploy/Synchronize Server Mapping and call the Deployment Wizard. See *Server Mapping Deployment Wizard* in the Designer documentation.

Undeploying Server-side Mapping Files from the RPC Server

Use the Server Mapping Deployment Wizard to undeploy a server-side mapping file (Designer file with extension .svm). See *Server Mapping Files for COBOL*.

➤ **To undeploy a server-side mapping file with the Server Mapping Deployment Wizard**

- 1 Make sure your RPC server is active and that the Deployment Service of the RPC server is properly configured. See [Deployment Service](#).
- 2 Make sure your IDL file is within a Designer directory (folder) without the related server-side mapping file (.svm).
- 3 From the context menu of your IDL file, choose **COBOL > Deploy/Synchronize Server Mapping** and call the Server Mapping Deployment Wizard. See *Server Mapping Deployment Wizard* in the Designer documentation. Because there is no related server-side mapping file in the Designer, all server mapping information related to the IDL file in the RPC server will be removed.

Change Management of Server-side Mapping Files

Under UNIX and Windows, change management for a directory or folder (server-side mapping container, see [Server-side Mapping Files in the RPC Server](#)) is similar to change management within ordinary operating system directories (folders). All updates to the directory or folder done after a backup must be kept.

All Designer server-side mapping files (.svm) added since the last backup should be available. See *Server Mapping Files for COBOL* in the Designer documentation.

List Deployed Server-side Mapping Files

Use the Windows Explorer or the UNIX `ls` command to list the contents of the server-side mapping container (directory or folder). See *Server-side Mapping Files in the RPC Server*.

Check if a Server-side Mapping File Revision has been Deployed

Server-side mapping files in the server-side mapping container correspond to lines of Designer files with extension .svm. See *Server Mapping Files for COBOL* in the Designer documentation. The files contain a creation timestamp at offset 276 (decimal) in the format *YYYYMMDDHHIISS*. Precision is 1/10 of a second. The creation timestamp can be checked.

The timestamp can be found on the same offset in the server-side mapping files stored in the server-side mapping container (directory or folder). See *Server-side Mapping Files in the RPC Server*.

Access Control: Secure Server Mapping File Deployment

For deployment with the *Server Mapping Deployment Wizard*, use EntireX Security if the broker is running on platforms z/OS, UNIX, Windows or z/VSE. See *Enabling the Deployment Service*.

Is There a Way to Smoothly Introduce Server-side Mapping Files?

All EntireX RPC servers can be executed without server-side mapping files. See *Server-side Mapping Files in the RPC Server*. There is no need to install the server-side mapping container if the following conditions are met:

- You do not use features that require server mapping; see *When is a Server Mapping File Required?*
- Server-side type of COBOL mapping is switched on in the Designer. If you have not used server-side mapping, we recommend you use client-side mapping. See *Server Mapping Files for COBOL*.

You can also call COBOL servers generated or extracted with previous versions of EntireX mixed with a COBOL server that requires server-side mapping. All EntireX RPC servers are backward compatible.

6 Scenarios

- COBOL Scenarios 36

COBOL Scenarios

Scenario I: Calling an Existing COBOL Server

➤ To call an existing COBOL server

- 1 Use the IDL Extractor for COBOL to extract the Software AG IDL and, depending on the complexity, also a server mapping file. See *When is a Server Mapping File Required?* in the Designer documentation.
- 2 Build an EntireX RPC client using any EntireX wrapper. For a quick test you can:
 - use the IDL Tester; see *EntireX IDL Tester* in the Designer documentation
 - generate an XML mapping file (XMM) and use the XML Tester for verification; see *EntireX XML Tester* in the XML/SOAP Wrapper documentation

See *Client and Server Examples for Micro Focus (UNIX and Windows)* in the COBOL Wrapper documentation for COBOL RPC Server examples.

Scenario II: Writing a New COBOL Server

➤ To write a new COBOL server

- 1 Use the COBOL Wrapper to generate a COBOL server skeleton and, depending on the complexity, also a server mapping file. See *When is a Server Mapping File Required?* in the Designer documentation. Write your COBOL server and proceed as described under *Using the COBOL Wrapper for the Server Side*.
- 2 Build an EntireX RPC client using any EntireX wrapper. For a quick test you can:
 - use the IDL Tester; see *EntireX IDL Tester* in the Designer documentation
 - generate an XML mapping file (XMM) and use the XML Tester for verification; see *EntireX XML Tester* in the XML/SOAP Wrapper documentation

See *Client and Server Examples for Micro Focus (UNIX and Windows)* in the COBOL Wrapper documentation for COBOL RPC Server examples.