webMethods EntireX

EntireX Micro Focus COBOL RPC Server

Version 9.7

October 2014

WEBMETHODS

This document applies to webMethods EntireX Version 9.7.

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

Document ID: EXX-MICROFOCUSRPC-97-20160805

Table of Contents

1 Inside the RPC Server 2
Inside the RPC Server 2
Usage of Server Mapping Files 5
2 Administering the Micro Focus RPC Server 7
Customizing the RPC Server 8
Configuring the RPC Server 9
Locating and Calling the Target Server 17
Using SSL or TLS with the RPC Server 18
Starting the RPC Server 19
Stopping the RPC Server 20
Activating Tracing for the RPC Server 21
3 Deployment Service 23
Introduction 24
Scope 25
Enabling the Deployment Service 25
Disabling the Deployment Service 26
4 Server-side Mapping Files 27
Server-side Mapping Files in the RPC Server 28
Deploying Server-side Mapping Files to the RPC Server 29
Undeploying Server-side Mapping Files to the RPC Server 29
Change Management of Server-side Mapping Files 30
List Deployed Server-side Mapping Files 30
Check if a Server-side Mapping File Revision has been Deployed 30
Access Control: Secure Server Mapping File Deployment 30
Is There a Way to Smoothly Introduce Server-side Mapping Files? 31
5 Running an EntireX RPC Server as a Windows Service 33
Introduction 34
Sample Installation 34
EntireX RPC Service Tool 36
Customizing the Service 38
Removing the Service 39
Tracing for the Windows Service 39
Starting an RPC Server Using the Service 39
Stopping an RPC Server Using the Service 40
Running with EntireX Security 40
Windows-specific Folders 41
6 Scenarios 43
COBOL Scenarios 44

1 Inside the RPC Server

- Inside the RPC Server 2
- Usage of Server Mapping Files 5

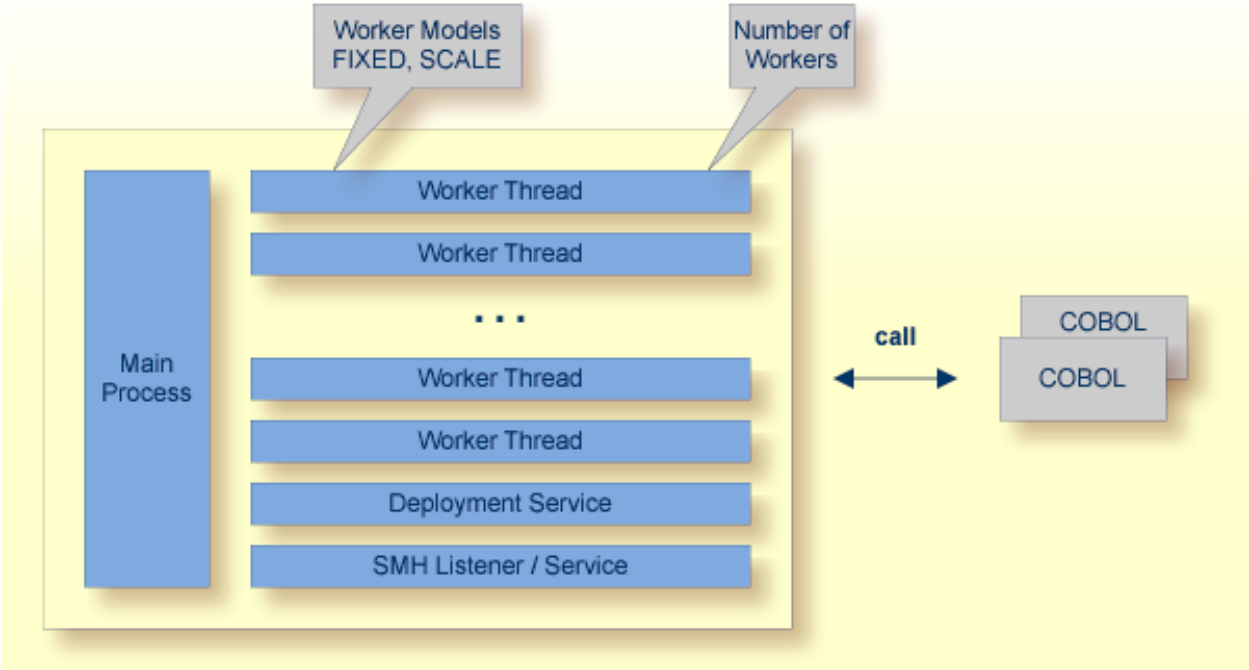
The EntireX Micro Focus COBOL RPC Server allows standard RPC clients to communicate with COBOL servers written with Micro Focus COBOL. It works together with the *COBOL Wrapper* and the *IDL Extractor for COBOL*.

Inside the RPC Server

- Worker Models

- Inbuilt Services

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 Micro Focus 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.
- `SCALE`
  The *scale* model creates worker threads depending on the incoming load of RPC requests.

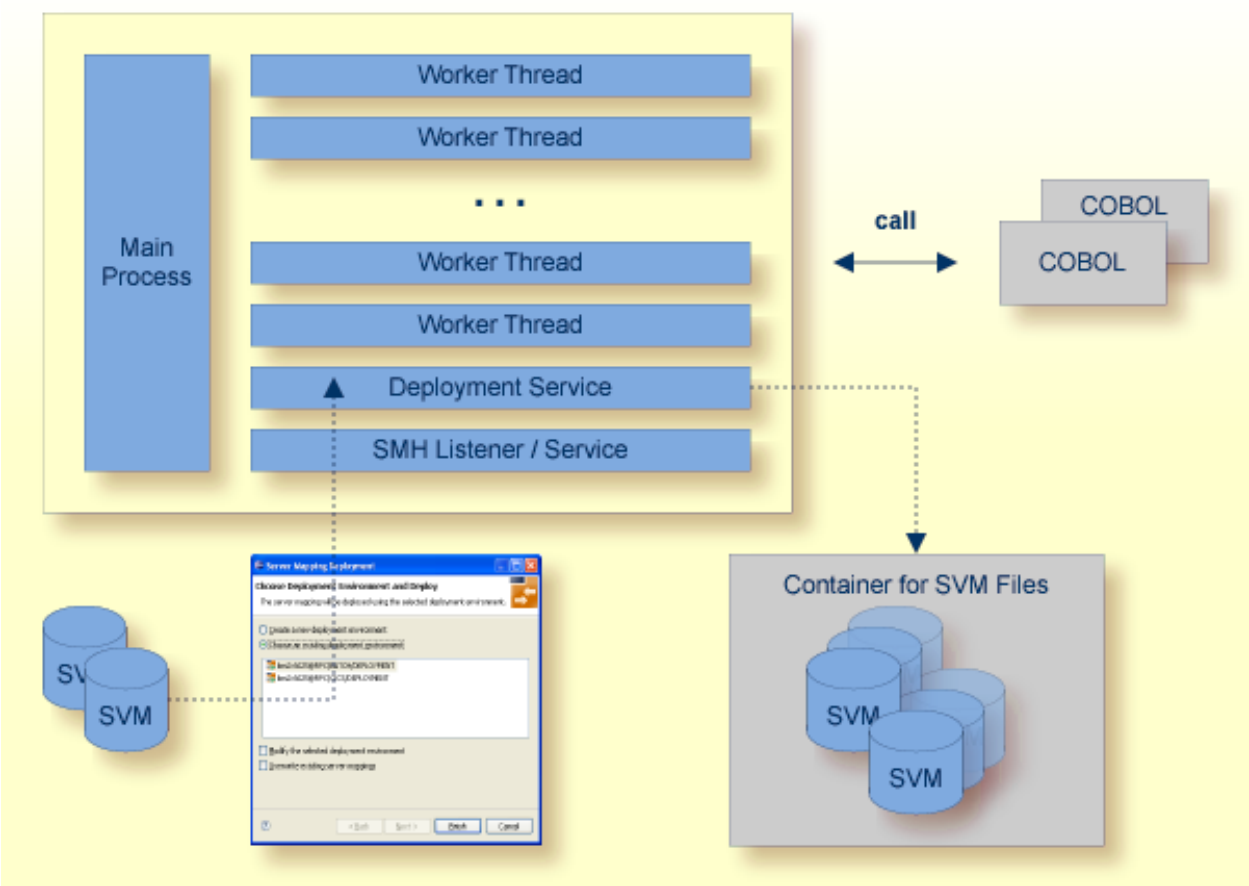
Inbuilt Services

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

- Deployment Service
- SMH Listener Service

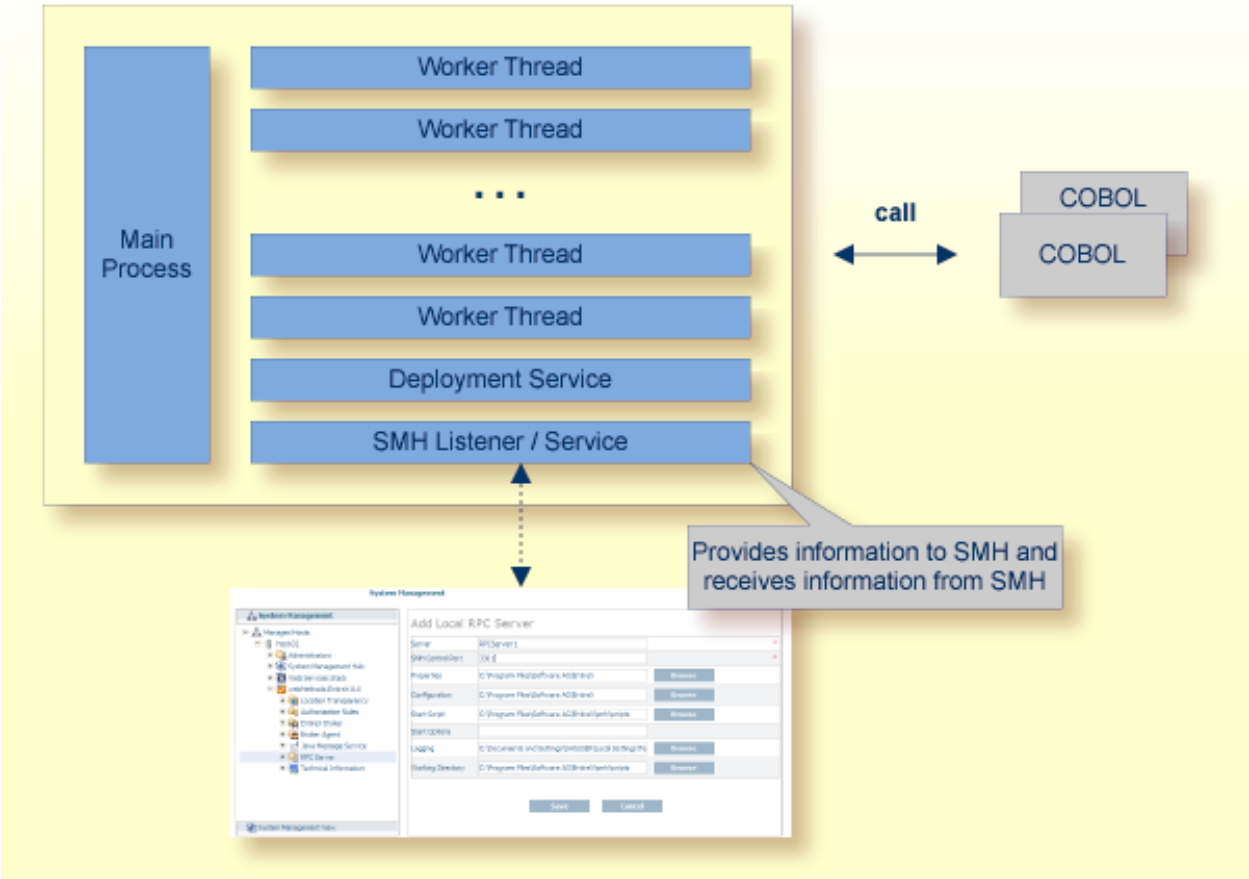
Deployment Service

The Deployment Service allows you to deploy server-side mapping files (EntireX Workbench 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.



SMH Listener Service

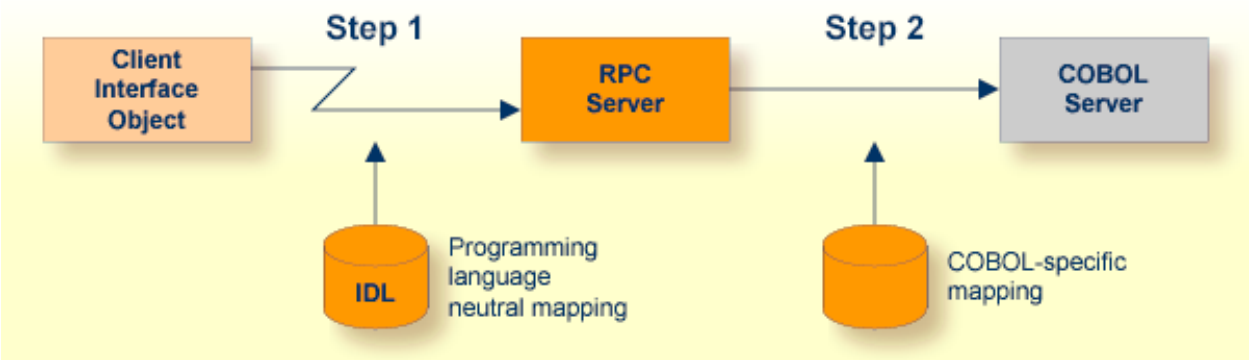
With the SMH Listener Service you use the System Management Hub to monitor the RPC server. See *Administering the EntireX RPC Servers using System Management Hub* in the UNIX and Windows administration documentation.



Usage of Server Mapping Files

There are many situations where the Micro Focus RPC Server 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 (*EntireX Workbench* files with extension .svm) and *client*-side mapping files (Workbench 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***.

Note: Server mapping files are used for COBOL only.

2 Administering the Micro Focus RPC Server

- Customizing the RPC Server 8
- Configuring the RPC Server 9
- Locating and Calling the Target Server 17
- Using SSL or TLS with the RPC Server 18
- Starting the RPC Server 19
- Stopping the RPC Server 20
- Activating Tracing for the RPC Server 21

The EntireX Micro Focus COBOL RPC Server 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 Micro Focus RPC Server:

- 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 Micro Focus RPC Server cannot be started and an error message is given.

Configuration File

The name of the delivered example configuration file is *microfocusserver.cfg*. The configuration file contains the configuration for the Micro Focus RPC Server. 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 name of the start script is platform-dependent:

- UNIX: *microfocusserver.bsh*
- Windows: *microfocusserver.bat*

The start script for the Micro Focus RPC Server contains the following:

- the 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, i.e. the commands/parameters `broker=localhost` and `brok=localhost` are equivalents.

Parameter	Default	Values	Req/ Opt
`brokerid`	`localhost`	Broker ID used by the server. See *Using the Broker ID in Applications* in the RPC Programming documentation. Example: `brokerid=myhost.com:1971`	R

Parameter	Default	Values	Req/ Opt
class	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 *Service-specific Attributes* (DEFAULTS=SERVICE) under *Broker Attributes* in the platform-independent administration documentation). Case-sensitive, up to 32 characters. Corresponds to CLASS. Example: class=MyRPC	R
codepage		Depending on the internationalization approach, the codepage (locale string) where incoming data is provided to the COBOL server. Conversely, the COBOL server must provide outgoing data in the given codepage, otherwise unpredictable results occur. See *What is the Best Internationalization Approach to use?* under *Internationalization with EntireX* for information on which internationalization approach requires a codepage (locale string). By default, no codepage is transferred to the broker. For the most popular internationalization approach, *ICU Conversion* under *Introduction to Internationalization*, the correct codepage (locale string) must be provided. This means it must: ■ follow the rules described under *Locale String Mapping* in the internationalization documentation ■ be a codepage supported by the broker ■ be the codepage used in your environment for file and terminal IO, otherwise unpredictable results may occur. Example: codepage=iso-8859-1	R/O
compresslevel	N	Enforce compression when data is transferred between broker and server. See *Data Compression in EntireX Broker* in the general administration documentation. compresslevel= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8| 9 | Y | N 0-9 0=no compression 9=max. compression N No compression.	O

Parameter	Default	Values	Req/ Opt
		Y Compression level 6. Example: compresslevel=6	
deployment	NO	Activates the deployment service, see ***Deployment Service***. Required to use the Server Mapping Deployment Wizard. See *Server Mapping Deployment Wizard* in the EntireX Workbench documentation. YES Activates the deployment service. The RPC server registers the deployment service in the broker. NO The deployment service is deactivated. The RPC server does not register the deployment service in the broker. Example: deployment=yes	O
encryptionlevel	0	Enforce encryption when data is transferred between client and server. Requires EntireX Security. See ENCRYPTION-LEVEL under *Broker ACI Fields*. 0 Encryption is enforced. 1 Encryption is enforced between server and broker kernel. 2 Encryption is enforced between server and broker kernel, and also between client and broker. Example: encryptionlevel=2	O
logon	YES	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 *Reliable RPC*. NO No logon/logoff functions are executed. YES Logon/logoff functions are executed. Example: logon=no	O

Parameter	Default	Values	Req/ Opt
marshalling	COBOL	The Micro Focus RPC Server supports COBOL. See also ***Locating and Calling the Target Server***. Marshalling=(LANGUAGE=COBOL, flavor=MF) 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 used to call the COBOL server correctly if one is available. See ***Usage of Server Mapping Files***.	O
password	no default	Password for broker logon. Case-sensitive, up to 32 characters. For more information see broker ACI control block field PASSWORD. Example: password=MyPwd	O
restartcycles	15	Number of restart attempts if the broker is not available. This can be used to keep the Micro Focus RPC Server running while the broker is down for a short time. A restart cycle will be repeated at an interval which is calculated as follows: timeout + ETB_TIMEOUT + 60 seconds where timeout is the RPC server parameter (see this table), and ETB_TIMEOUT is the environment variable (see *Environment Variables in EntireX* in the general administration documentation) When the number of cycles is reached and a connection to the broker is not possible, the RPC server stops. Example: restartcycles=30	O
servername	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 *Service-specific Attributes* (DEFAULTS=SERVICE) under *Broker Attributes* in the platform-independent administration documentation. Case-sensitive, up to 32 characters. Corresponds to SERVER of the broker attribute file. Example: servername=mySrv	R

Parameter	Default	Values	Req/ Opt
service	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 *Service-specific Attributes* (DEFAULTS=SERVICE) under *Broker Attributes* in the platform-independent administration documentation. Case-sensitive, up to 32 characters. Corresponds to SERVICE attribute of the broker attribute file. Example: service=MYSERVICE	R
smhport	0	The port where the server listens for commands from the System Management Hub (SMH). If this port is 0 (default), no port is used and management by the SMH is disabled. Example: smhport=3001	O
ssl_file	no default	Set the SSL parameters. See ***Using SSL or TLS with the RPC Server*** for examples and more information.	O
svm		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=*path*) *path* 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
timeout	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. Example: timeout=300	O

Parameter	Default	Values	Req/ Opt
tracedestination	ERXTrace.*nnn*.log	The name of the destination file for trace output. By default the main trace file name is ERXTrace.*nnn*.log, where *nnn* can be in the range from 001 to 005. UNIX: The trace file is located in the current working directory. Windows: The trace file is located in a subfolder of the windows folder My Documents. Example: tracedestination=ERXTrace.log	O
tracelevel	None	Trace level for the server. See also ***Activating Tracing for the RPC Server***. tracelevel = None | Standard | Advanced ↩ | Support None: No trace output. Standard: For minimal trace output. Advanced: For detailed trace output. Support: This trace level is for support diagnostics and should only be switched on when requested by Software AG support. Example: tracelevel=standard	O
traceoption	None	Additional trace option if trace is active. None: No additional trace options. STUBLOG: If tracelevel is Advanced or Support, the trace additionally activates the broker stub log. 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. **Note:** This can increase the amount of trace output data dramatically if you transfer large data buffers. Example: traceoption=(STUBLOG,NOTRUNC)	O

<table>
<tr><th>Parameter</th><th>Default</th><th>Values</th><th>Req/ Opt</th></tr>
<tr><td>userid</td><td>ERX-SRV</td><td>Used to identify the server to the broker. See broker ACI control block field USER-ID. Case-sensitive, up to 32 characters.

Example:
userid=MyUid</td><td>R</td></tr>
<tr><td>workermodel</td><td>SCALE,1,3,slowshrink</td><td>The Micro Focus RPC Server can be configured to

■ adjust the number of worker threads to the current number of client requests:

workermodel=(SCALE,from,thru
[,slowshrink ↩
| fastshrink]
[,noisolation | ↩
isolation])

■ use a fixed number of worker threads:

workermodel=(FIXED,number
[,noisolation ↩
| isolation])

<table>
<tr><td>FIXED</td><td>A fixed number of worker threads is used by the Micro Focus RPC Server.</td></tr>
<tr><td>SCALE</td><td>The number of worker threads is adjusted to the current number of client requests. With the from value, the minimum number of active worker threads can be set. The thru value restricts the maximum number of worker threads.
<table>
<tr><td>slowshrink</td><td>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. This is the default if SCALE is used.</td></tr>
<tr><td>fastshrink</td><td>The RPC server stops worker</td></tr>
</table></td></tr>
</table></td><td>O</td></tr>
</table>

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

Locating and Calling the Target Server

Introduction

The Micro Focus RPC Server is able to call standard libraries (Windows DLLs or UNIX so|sl); 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 Micro Focus RPC Server 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 or TLS with the RPC Server

There are two ways of specifying SSL or TLS, depending on the complexity of the parameters:

- as part of the Broker ID for short parameters, the simplest way
- using the SSL file, a text file containing more complex parameters.

For more information, see *SSL or TLS and Certificates with EntireX*.

Specifying the SSL or TLS Parameters as Part of the Broker ID

The simplest way to specify SSL or TLS parameters is to add them to the Broker ID.

Example:

```
ssl://ETB001?TRUSTSTORE=whatever
```

Specifying the SSL or TLS Parameters in a Separate File

For complex SSL or TLS parameters there is the SSL file, a text file containing the parameters.

The `SSL_FILE` keyword points to this text file.

To specify the SSL or TLS parameters in the SSL file

1 Set the parameters as described under *Running Broker with SSL or TLS Transport* in the platform-specific administration documentation.

2 Prefix/suffix the Broker ID with the SSL key.

Example:

```
brokerid=SSL://ETB001
.
.
ssl_file=C:\mySSLdirectory\mySSLParms.txt
```

Starting the RPC Server

To start the Micro Focus RPC Server

- Use the script *microfocusrpcserver* in the folder *bin* to start the Micro Focus RPC Server. You may customize this file.

  Or:

  Use the RPC server agent in the System Management Hub to configure and start the Micro Focus RPC Server.

  See *Administering the EntireX RPC Servers using System Management Hub* in the UNIX and Windows administration documentation for details.

  Or:

Use the Micro Focus RPC Server as a Windows service, see *Running an EntireX RPC Server as a Windows Service* in the Micro Focus RPC Server documentation.

Stopping the RPC Server

To stop the Micro Focus RPC Server

- Use the RPC server agent in the SMH to stop the Micro Focus RPC Server.

  Or:

  Use the agent for the broker. Use `Deregister` on the service, specified with the parameters class/servername/service.

Activating Tracing for the RPC Server

To switch on tracing for the RPC server

- Set the parameters `tracelevel` and `tracedestination`. See ***Configuring the RPC Server***.

  To evaluate the return codes, see *Error Messages and Codes*. See also *Tracing the RPC Server* in the UNIX and Windows administration documentation.

3 Deployment Service

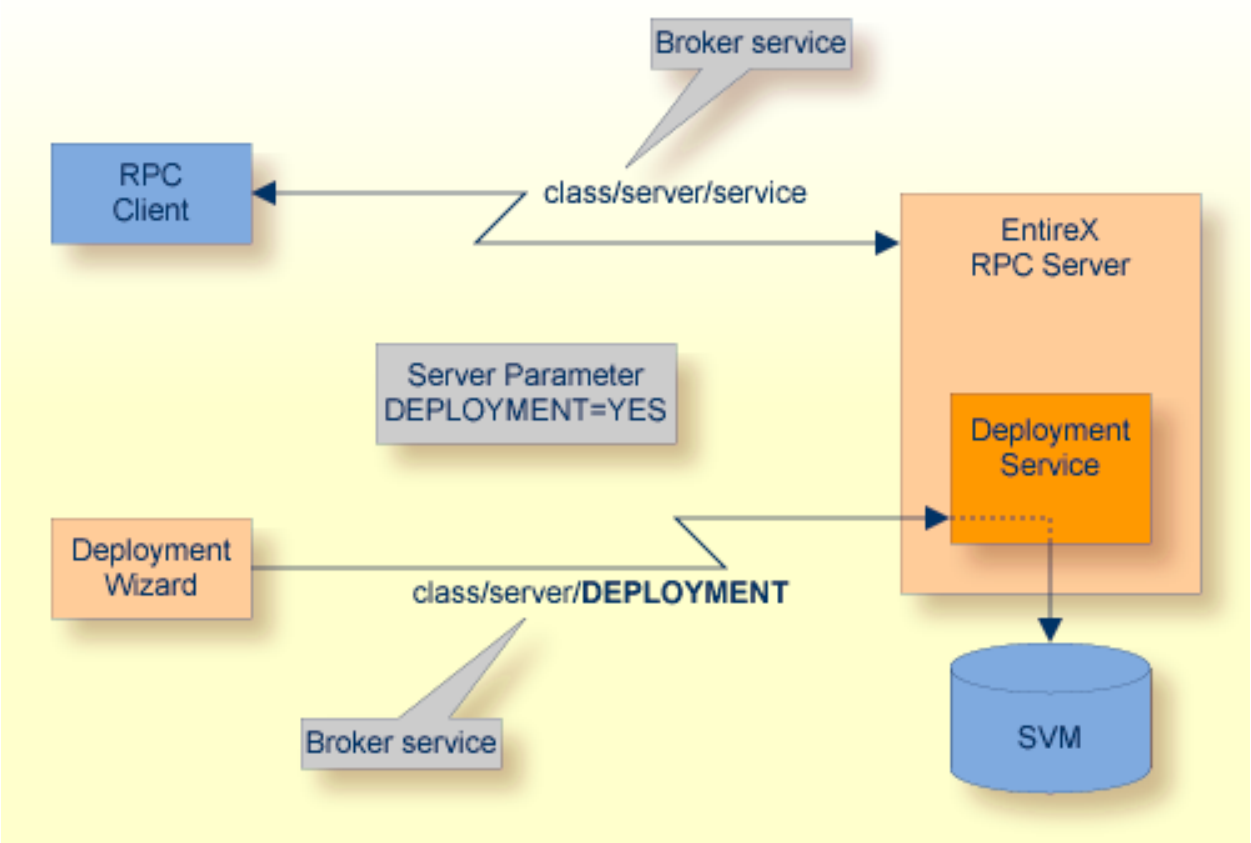
- Introduction 24
- Scope 25
- Enabling the Deployment Service 25
- Disabling the Deployment Service 26

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* under *Overview of EntireX Security* 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 EntireX Workbench 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 a Micro Focus RPC Server, configure the server mapping file subparameter `path` of parameter `svm` to point to a directory with write access. See *Configuring the RPC Server* under *Administering the Micro Focus 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* in the EntireX Security documentation.

- For a UNIX or Windows broker, see *Administering Authorization Rules using System Management Hub* in the UNIX and Windows administration documentation.
- Not applicable to a BS2000/OSD or z/VSE broker.

Disabling the Deployment Service

To disable the deployment service

- Set the Micro Focus RPC Server parameter `deployment=no`. See `deployment` under ***Configuring the RPC Server***.

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

4 Server-side Mapping Files

- Server-side Mapping Files in the RPC Server 28
- Deploying Server-side Mapping Files to the RPC Server 29
- Undeploying Server-side Mapping Files to the RPC Server 29
- Change Management of Server-side Mapping Files 30
- List Deployed Server-side Mapping Files 30
- Check if a Server-side Mapping File Revision has been Deployed 30
- Access Control: Secure Server Mapping File Deployment 30
- Is There a Way to Smoothly Introduce Server-side Mapping Files? 31

Server mapping enables the RPC server to correctly support special COBOL syntax such as `REDEFINE`s, `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 (EntireX Workbench files with extension .cvm) and server-side mapping files (Workbench 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 EntireX Workbench 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 EntireX Workbench documentation.

Server-side Mapping Files in the RPC Server

For Micro Focus RPC Servers under UNIX or Windows, server-side mapping corresponds to lines of EntireX Workbench 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* under *Administering the Micro Focus 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 (Workbench file with extension .svm) with the Server Mapping Deployment Wizard. See *Server Mapping Files for COBOL* in the EntireX Workbench 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 EntireX Workbench documentation.

Undeploying Server-side Mapping Files to the RPC Server

Use the Server Mapping Deployment Wizard to undeploy a server-side mapping file (Workbench 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 an EntireX Workbench 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 EntireX Workbench documentation. Because there is no related server-side mapping file in the Workbench, 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 EntireX Workbench server-side mapping files (.svm) added since the last backup should be available. See *Server Mapping Files for COBOL* in the EntireX Workbench documentation.

List Deployed Server-side Mapping Files

Use the Windows Explorer (for Windows) or the `ls` command (for UNIX) 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 EntireX Workbench files with extension .svm. See *Server Mapping Files for COBOL* in the EntireX Workbench documentation. The files contain a creation timestamp at offset 276 (decimal) in the format *YYYYMMDDHHIISST*. 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 EntireX Workbench. 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.

5 Running an EntireX RPC Server as a Windows Service

- Introduction 34
- Sample Installation 34
- EntireX RPC Service Tool 36
- Customizing the Service 38
- Removing the Service 39
- Tracing for the Windows Service 39
- Starting an RPC Server Using the Service 39
- Stopping an RPC Server Using the Service 40
- Running with EntireX Security 40
- Windows-specific Folders 41

Introduction

Any EntireX RPC server can be run as a Windows service. This has the following advantages:

- The server is started automatically when the PC is booted and shut down automatically when the PC is shut down.
- The service can be run under a system account that has different rights on a PC than the user account.
- The service can be run under any valid user ID.
- The Control Panel provides an overview of started servers and their status.

To run an RPC server as a Windows service, a batch script (.bat) that can start the RPC server is required. Sample batch scripts are provided in the EntireX *bin* folder:

- microfocusserver.bat
- cserver.bat
- cicseciserver.bat
- imsconnectserver.bat
- dotNetServer.bat
- jrpcserver.bat
- jxmlrpcserver.bat
- ...

To install RPC server as windows services, use the EntireX RPC Service Tool. Multiple services can be installed if more than one RPC server is required.

Sample Installation

Note: The steps below use the C RPC Server as an example, but the information applies to all RPC servers.

To install an RPC Server as a Windows Service

1 Enter the EntireX installation directory and copy file *EntireX\bin\RPCService.bat*. Save the file under a different name, for example *EntireX\bin\RPCService_C.bat*.

The EntireX installation directory is specified during installation. Later on when registering the service, the argument value of parameter `-ext` of `rpcservice.exe` must correspond with this extension, for example `-ext C`.

The extension `_C` will later correspond with the argument in parameter –ext when registering the service using rpcservice.exe. Also the name of the service will be extended by this value. This is needed to distinguish multiple server instances.

2 Modify batch file *RPCService_C.bat* and uncomment the line where the C RPC server batch file is referenced.

```
cserver.bat %*
```

The file contains entries for all the different types of RPC servers. The C RPC server is already uncommented, it is the default.

3 Adapt the corresponding configuration file, for example *\EntireX\config\cserver.cfg*, to your needs.

4 Optionally test whether the server is configured correctly before registering it as a Windows service by entering the following from a command prompt:

```
RPCService_C.bat
```

5 Register the Server as a Windows service. Enter:

```
rpcservice.exe -install -ext C -serverlog c:\serverlog.txt -trace
c:\servertrace.txt -script c:\SoftwareAG\EntireX\bin\RPCservice_C.bat
```

where `c:\SoftwareAG` is the installation directory,

`-ext`	must correspond with the extension specified in step 1
`-serverlog`	is optional
`-trace`	is optional
`-script`	must point to file *RPCservice_C.bat* created in step 1

EntireX RPC Service Tool

The EntireX RPC Service Tool is provided to install, remove, start and stop RPC services. An overview screen “Software AG EntireX RPC Service” lists the available arguments and options.

Displaying the available Arguments and Options

To display the overview screen

- In the command line, go to the EntireX *bin* directory and enter:

```
rpcservice.exe -help
```

The following screen is displayed:

```
            Software AG EntireX RPC Service (V.n.n.n)
             Service Tool to start/stop RPC servers
      (c) Copyright Software AG 1997-20nn. All rights reserved.
           ------------------------------------------------

Usage:
rpcservice -install         [options]    to install the service
rpcservice -remove          [options]    to remove the service
rpcservice -help            [options]    prints this panel

Options:
      [-ext[ension]     <extension>]         use a service extension
      [-script          <path\script>]       set the server startup file
                                             for this service
      [-serverlog       <path\logfile>]      set the communication file
                                             between server and service
      [-userid          <userid>]            set userid for broker logon
      [-password        <password>]          set password for broker logon
      [-trace           [<tracefile>]]       used for diagnostics
```

Explanation of Options

- **No option**
  This will install a service displayed as “Software AG EntireX RPC Service”. The service is registered as EXXRPCService (Service name under Properties). Required suboptions will be set to their defaults.
- **-extension** *<ext>*
  use this option to add an extension to the service to differentiate it from other instances, if you want to install multiple RPC service instances. The service will be displayed as “Software AG EntireX RPC Service [<ext>]”. The default log file *RPCservice_<ext>.log.* will be added.

  Important: If you use this option, the program will by default search for a script with the same extension, *RPCService_<ext>.bat*. Make sure it exists.

  Example:

  - For a Java RPC Server (see also *Running the Java RPC Server as a Windows Service* under under *Administering the EntireX Java RPC Server* in the Windows administration documentation in the Windows administration documentation), you can use

    ```
    -install -ext java -script <EntireX bin directory>\jrpcserer.bat
    ```

    The service will be displayed as Software AG EntireX RPC Service [java], the log file will be called *RPCservice_java.log*.
  - For a Micro Focus RPC Server you can use:

    ```
    -install -ext MicroFocus -script  <EntireX bin directory>\microfocusserver.bat
    ```

    The service will be displayed as Software AG EntireX RPC Service [MicroFocus], the log file will be called *RPCservice_MicroFocus.log*.
  - For a C RPC Server you can use:

    ```
    -install -ext C -script cserver.bat
    ```

    The service will be displayed as Software AG EntireX RPC Service [C], the log file will be called *RPCservice_C.log*.

  Default: none.
- **-script**
  use this option to specify a batch script other than the default scripts *RPCService.bat* or *RPCService_<ext>.bat*.

  Important: The script file must pass external parameters to the RPC server. Please make sure that %* remains at the end of the line that contains the RPC server call, so that parameters coming from the Windows Service are passed to the RPC server:

```
rpcserver.exe "cfg=..\config\cserver.cfg"%*
```

Default: <EntireX-installation>\bin\RPCService.bat
or, if -ext <ext> is applied:
<EntireX-installation>\bin\RPCService_<ext>.bat

- **-serverlog**
  We recommend that you do not change the name of this file.

  Default: CSIDL_LOCAL_APPDATA\Software AG\EntireX\RPCservice.log
  or, if -ext <ext> is applied:
  CSIDL_LOCAL_APPDATA\Software AG\EntireX\RPCservice_<ext>.log

  CSIDL_LOCAL_APPDATA is resolved at runtime according Microsoft's policy for user and application repositories. See *Windows-specific Folders*.

- **-userid and -password**
  use these options if your target Broker requires EntireX Security.

  These options are obsolete from EntireX versions 8.2 SP1. RPC servers communicate their user ID/password credentials via the server log file if required by EntireX Security. See *Running with EntireX Security*. The options are still supported for compatibility with earlier versions or if the user ID and password of the RPC service have to be different from those of the RPC server(s). If the user ID and password are applied in this RPC service (and are different from the default), these values take precedence over the credentials that may be supplied by the RPC server(s).

  Defaults:
  User ID: ERX-USER
  Password: PASSWORD

  Note: This service requires the broker.dll. Make sure that when the service is executing under the control of the Windows Service Manager it can access the broker.dll in the standard path definitions or in the EntireX installation directory.

Customizing the Service

You can set the service to start the RPC server automatically or manually.

To customize the service

1. Go to **Control Panel** > **Administrative Tools** > **Services** and select the service and then **Properties**.
2. Set the Startup type as desired or
3. Start/stop the service as desired.

Removing the Service

To remove the service

- In the command line, go to the EntireX *bin* folder and enter:

```
rpcservice.exe -remove [-ext[ension] <extension>]
```

Tracing for the Windows Service

Use the `-trace` option with the individual commands.

- with `-install`

  The trace option will be stored in the System Registry and tracing will be enabled if the service is started/stopped by the Service Manager. It can only be switched off by removing the service.
- With the other commands trace runs as long as the command takes to finish its task.
- You can specify a target log file. By default the file is written to the Windows folder for user documents. See *Folder for User Documents*.

Starting an RPC Server Using the Service

If the startup type for the service is set to automatic, the server will be started when the PC is booted.

To start the server manually

- Go to **Control Panel** > **Administrative Tools** > **Services** and select the service, then **Properties** and choose **Start**.

Stopping an RPC Server Using the Service

If you selected startup type automatic in the service properties, the server will be stopped by the operating system when the PC is shut down.

To stop the server manually

- Use the service, go to **Control Panel** > **Administrative Tools** > **Services** and select the service, then **Properties** and choose the **Stop** button.

If you choose to stop the server using the EntireX utilities, the Windows service will change the status from "running" to "stopped" and writes a warning message in the event log that the EntireX server has stopped for an unknown reason.

Running with EntireX Security

Starting an RPC Server running under EntireX Security

To start an RPC server, supply a valid user ID and password to log on to the broker. The ID and password are applied to the configuration of the individual RPC server(s).

Stopping an RPC Server running under EntireX Security;

When you stop an RPC server, the RPC service uses the user ID/password from the credentials provided by the new RPC server(s), if applied. Otherwise user ID and password may be taken from the RPC service installation options (see above).

Note: Broker command service `etbcmd` is used to stop the broker. See *Broker Command and Information Services*. This means that users who can start RPC servers to certain broker services also need access rights to `Class=SAG, Server=ETBCIS, Service=*`.

Windows-specific Folders

Folder for Application Data

The file system directory that serves as a common repository for application-specific data.

A typical path is *C:\Documents and Settings\username\Application Data*.

Under Windows 7, this path for the SYSTEM user might resolve to:
"C:\Windows\System32\config\systemprofile\AppData\Local".

Folder for Local (Non-roaming) Application Data

The file system directory that serves as a data repository for local (non-roaming) applications.

A typical path is *C:\Documents and Settings\username\Local Settings\Application Data*.

Folder for User Documents

The file system directory used to physically store a user's common repository of documents.

A typical path is *C:\Documents and Settings\username\My Documents*.

See **MSDN Library**.

6 Scenarios

- COBOL Scenarios 44

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 EntireX Workbench 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 EntireX Workbench 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 EntireX Workbench 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 EntireX Workbench 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.