**software** AG

# webMethods EntireX

## CICS ECI RPC Server

Version 9.7

October 2014

**WEBMETHODS**

# Table of Contents

# 1 Introduction to the CICS ECI RPC Server

The EntireX CICS® ECI RPC Server allows standard RPC clients to communicate with CICS programs running on IBM CICS® version 3.2 and higher. The CICS ECI RPC Server transforms the RPCs from the clients into messages to CICS ECI. The CICS ECI RPC Server acts on one side as an RPC server and on the other side as a client for CICS ECI. The CICS ECI RPC Server is a Java-based component that can run on a different host to the one where CICS is running. This allows it to operate with a zero footprint of EntireX on the CICS host.
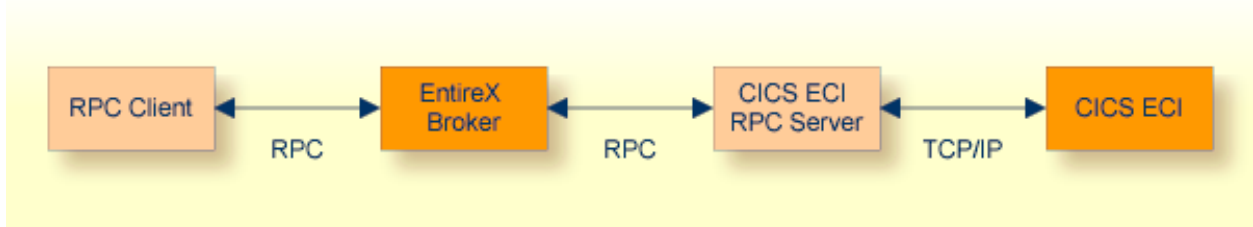


For existing COBOL programs you can use the *Software AG IDL Extractor for COBOL* to generate the IDL file for the RPC clients. The list under `DATA DIVISION` *Mapping* under *COBOL to IDL Mapping* in the IDL Extractor for COBOL documentation discusses which clauses of COBOL syntax are supported and how these are handled by the extractor.

For existing PL/I programs you can use the *Software AG IDL Extractor for PL/I* to generate the IDL file for the RPC clients.

All source files, COBOL or PL/I, have to exist locally for the CICS ECI RPC Server. Remote extraction is not possible if there is no EntireX RPC server (batch or IMS) with extractor service enabled on the CICS host.

The CICS ECI RPC Server supports RPC clients in different programming languages.

# 2 Administrating EntireX CICS® ECI RPC Server

The EntireX CICS® ECI RPC Server allows standard RPC clients to communicate with CICS programs running on IBM CICS® version 3.2 and higher. The CICS ECI RPC Server transforms the RPCs from the clients into messages to CICS ECI. The CICS ECI RPC Server acts on one side as an RPC server and on the other side as a client for CICS ECI. The CICS ECI RPC Server is a Java-based component that can run on a different host to the one where CICS is running. This allows it to operate with a zero footprint of EntireX on the CICS host.

# Customizing the CICS ECI RPC Server

For the setup of the CICS ECI RPC Server there are

- a configuration file and
- scripts to start the CICS ECI RPC Server.

### Location of the CICS ECI RPC Server

The CICS ECI RPC Server is contained in the file *entirex.jar*.

### The Configuration File

The default name of the configuration file is *entirex.cicseci.properties*. The CICS ECI RPC Server searches for this file in the current working directory.

You can set the name of the configuration file with `-Dentirex.server.properties=<your file name>` with "/" as file separator.

The configuration file contains the configuration for both parts of the CICS ECI RPC Server.



EntireX CICS ECI RPC Server Configuration
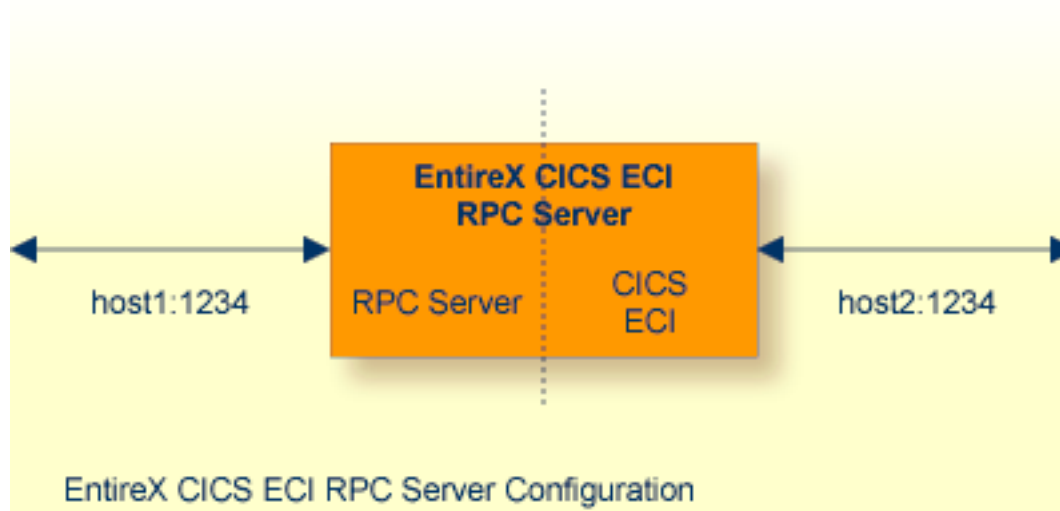
≫ **To set up the CICS ECI RPC Server**

1     Use the RPC server agent of the System Management Hub.

2     Add the CICS ECI RPC Server as an RPC server.

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

      Or:

      Use the scripts to start the CICS ECI RPC Server.

      Both scripts use the configuration file *entirex.cicseci.properties* in the folder *etc*.

## Configuring more than one CICS ECI RPC Server

If you configure more than one CICS ECI RPC Server that connect to the same EntireX Broker, the
following items must be distinct:

- the trace output file (property `entirex.server.logfile`)
- the monitor port for SMH (property `entirex.server.monitorport`)
- the log for the Windows Service (property `entirex.server.serverlog`)
- the trace output file of the SMH agent for RPC servers

# Configuring the RPC Server Side

The RPC server side of the CICS ECI RPC Server is configured like the Java RPC Server. The CICS ECI RPC Server uses the properties that start with "entirex.server".

The RPC server side can adjust the number of worker threads to the number of parallel requests. Use the properties `entirex.server.fixedservers`, `entirex.server.maxservers` and `entirex.server.minservers` to configure this scalability.

- If `entirex.server.fixedservers=yes`, the number of `entirex.server.minservers` is started and the server can process this number of parallel requests.

- If `entirex.server.fixedservers=no`, the number of worker threads balances between `entirex.server.minservers` and `entirex.server.maxservers`. This is done by a so-called attach server thread. On startup, the number of worker threads is `entirex.server.minservers`.

If more than `entirex.server.minservers` are waiting for requests, a worker thread stops if its receive call times out. The timeout period is configured with `entirex.server.waitserver`.

Alternatively to the properties, you can use the command-line option. The command-line options have a higher priority than the properties set as Java system properties and these have higher priority than the properties in the configuration file.

| Name | Command-line Option | Default Value | Explanation |
|------|---------------------|---------------|-------------|
| `entirex.bridge.verbose` | | `no` | Verbose/trace mode of CICS ECI RPC Server. Set this to `"yes"` to trace sent to CICS ECI. |
| `entirex.server.brokerid` | `-broker` | `localhost` | Broker ID |
| `entirex.server.serveraddress` | `-server` | `RPC/SRV1/CALLNAT` | Server address. |
| `entirex.server.userid` | `-user` | `CICSECIRPCServer` | The user ID for the Broker for RPC. See `entirex.server.password`. |
| `entirex.server.fixedservers` | | `no` | `no` Use attach server to manage worker threads. `yes` Run minimum number of server threads. |
| `entirex.server.minservers` | | `1` | Minimum number of server threads. |
| `entirex.server.maxservers` | | `32` | Maximum number of server threads. |
| `entirex.server.restartcycles` | `-restartcycles` | `15` | Number of restart attempts if the Broker is not available. This can be used to keep the CICS ECI RPC Server running while the Broker is down for a short time. |

| Name | Command-line Option | Default Value | Explanation |
|---|---|---|---|
| `entirex.server.password` | `-password` | | The password for secured access to th Broker. The password is encrypted and writte the property `entirex.server.password.e`. To change the password, set the new password in the properties file (defau `entirex.cicseci.properties`). To disable password encryption set `entirex.server.passwordencryp` Default for this property is "yes". |
| `entirex.server.security` | `-security` | `no` | no\|yes\|auto\|name of `BrokerSecuri` object |
| `entirex.server.encryptionlevel` | | `0` | Encryption level. Valid values: 0,1,2. |
| `entirex.server.compresslevel` | `-compresslevel` | `0` | Permitted values (you can enter the te the numeric value)<br><br>`BEST_COMPRESSION     9`<br>`BEST_SPEED           1`<br>`DEFAULT_COMPRESSION -1`, mapped<br>`DEFLATED             8`<br>`NO_COMPRESSION       0`<br>`N                    0`<br>`Y                    8` |
| `entirex.server.waitattach` | | `600S` | Wait timeout for the attach server thre |
| `entirex.server.waitserver` | | `300S` | Wait timeout for the worker threads. |
| `entirex.timeout` | | `20` | TCP/IP transport timeout. See *Setting Transport Timeout* under *Writing Adva Applications - EntireX Java ACI*. |
| `entirex.server.verbose` | `-verbose` | `no` | Enable verbose output to the log file. |
| `entirex.server.logfile` | `-logfile` | | Name of the log file, default is standa output. |
| `entirex.trace` | `-trace` | `0` | Trace level (1,2,3). |
| `entirex.server.monitorport` | `-smhport` | `0` | The port where the server listens for commands from the System Manager Hub (SMH). If this port is 0, no port is and management by the SMH is disab |

## Configuring the CICS ECI Side

These properties are used to configure the connection to CICS ECI.

Alternatively, you can use the command-line option. The command-line options have a higher priority than the properties set as Java system properties and these have higher priority than the properties in the configuration file.

| Name | Default Value | Explanation |
|---|---|---|
| cics.host | | Host name of CICS ECI. Mandatory. |
| cics.port | | Port number of CICS ECI. Mandatory. |
| cics.transaction | | Name of the CICS mirror transaction that will receive transactions. Mandatory. |
| entirex.bridge.targetencoding | cp037 | Specify the appropriate EBCDIC encoding used by your CICS ECI. This codepage is also used when communicating with the EntireX Broker.<br><br>**Note:** Enable conversion in the Broker attribute file so the data can be converted correctly, typically by setting service-specific attribute CONVERSION to "SAGTCHA".<br><br>Default "cp037" is EBCDIC codepage with full Latin-1 character set. |
| cics.sockettimeout | 10000 | Socket timeout for connection to CICS ECI (in milliseconds). |
| cics.userid | | RACF user ID. Maximum 8 bytes (optional). |
| cics.password | | RACF password/PassTicket. Maximum 8 bytes (optional). |
| cics.sslparams | | SSL parameters (optional). Same syntax as Broker ID. |
| cics.mapping.folder | | The folder where the RPC server expects server-side mapping files (EntireX Workbench files with extension .svm). See *Deploying Server-side Mapping Files to the RPC Server* and *Undeploying Server-side Mapping Files to the RPC Server*.<br><br>There are also client-side mapping files that do not require configuration here. See *Server Mapping Files for COBOL*.<br><br>If *no* server requires server-side mapping, you can omit this property.<br><br>If *one* server requires server-side mapping, this property must be specified. |

## Starting the CICS ECI RPC Server

### ≫ To start the CICS ECI RPC Server

■   Use the script *cicseciserver* in the folder *bin* to start the CICS ECI RPC Server. You may customize this file.

Or:

Use the RPC server agent in the System Management Hub to configure and start the CICS ECI RPC Server.

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

## Stopping the CICS ECI RPC Server

### ≫ To stop the CICS ECI RPC Server

■   Use the RPC server agent in the SMH to stop the CICS ECI RPC Server.

Or:

Use the agent for the Broker. Use `Deregister` on the service, specified with the property `entirex.server.serveraddress`.

# Application Identification

The application identification is sent from the CICS ECI RPC Server to the Broker. It is visible with Broker Command and Information Services.

The identification consists of four parts: name, node, type, and version. These four parts are sent with each Broker call and are visible in the trace information.

For the CICS ECI RPC Server, these values are:

| Identification Part | Value |
|---|---|
| Application name | `ANAME=CICS ECI RPC Server` |
| Node name | `ANODE=<host name>` |
| Application type | `ATYPE=Java` |
| Version | `AVERS=9.5.0.0` |

# 3 Server-side Mapping Files

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 a CICS ECI RPC Server, server mapping information is contained in a server-side mapping file (EntireX Workbench file with extension .svm) See *Server Mapping Files for COBOL*. Server mapping files are provided as operating system files in an RPC server related server-side mapping container (directory or folder). The files have the same format as in the Workbench. See *Configuring the CICS ECI Side*.

If *no* server requires a server mapping file, you can omit the property `cics.mapping.folder`.

If *one* server requires a server mapping file, provide the property `cics.mapping.folder`.

See also *Deploying Server-side Mapping Files to the RPC Server*.

## Deploying Server-side Mapping Files to the RPC Server

Deploy a server-side mapping file (Workbench file with extension .svm) to the RPC server manually. See *Server Mapping Files for COBOL* in the EntireX Workbench documentation.

≫ **To deploy a server-side mapping file**

1   Make sure the server-side mapping container (directory or folder) is configured. See *Server-side Mapping Files in the RPC Server*.

2   Copy the server-side mapping file to the server-side mapping container.

## Undeploying Server-side Mapping Files to the RPC Server

Undeploy a server mapping file (Workbench file with extension .svm) from the RPC server manually. See *Server Mapping Files for COBOL*.

≫ **To undeploy a server-side mapping file manually**

■    Delete the server-side mapping file from the server-side mapping container (directory or folder). See *Server Mapping Files for COBOL*.

## 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 EntireX Workbench files with extension .svm (same format). See *Server Mapping Files for COBOL* in the EntireX Workbench documentation. Each line relates to an IDL program and contains 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*.

# 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.

# 4    **Scenarios**

# 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 *Basic RPC Server Examples - CALC, SQUARE* and *Reliable RPC Server Example - SENDMAIL* 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 *Basic RPC Server Examples - CALC, SQUARE* and *Reliable RPC Server Example - SENDMAIL* in the COBOL Wrapper documentation for COBOL RPC Server examples.