webMethods EntireX

CICS ECI RPC Server

Version 9.6

April 2014

This document applies to webMethods EntireX Version 9.6.

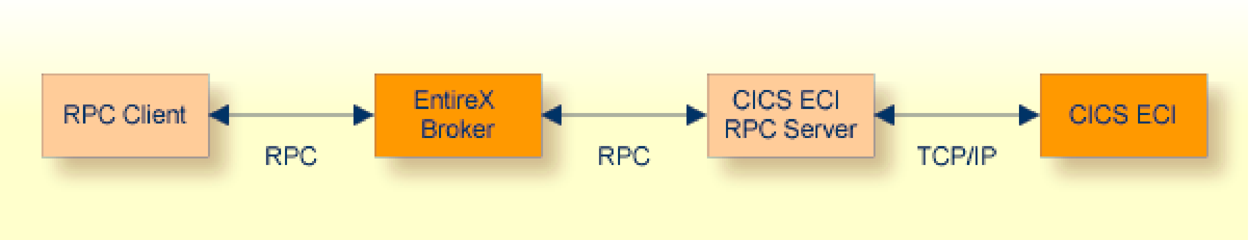
Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: EXX-CICSECI-96-20140628

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 COBL 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 the CICS ECI RPC Server

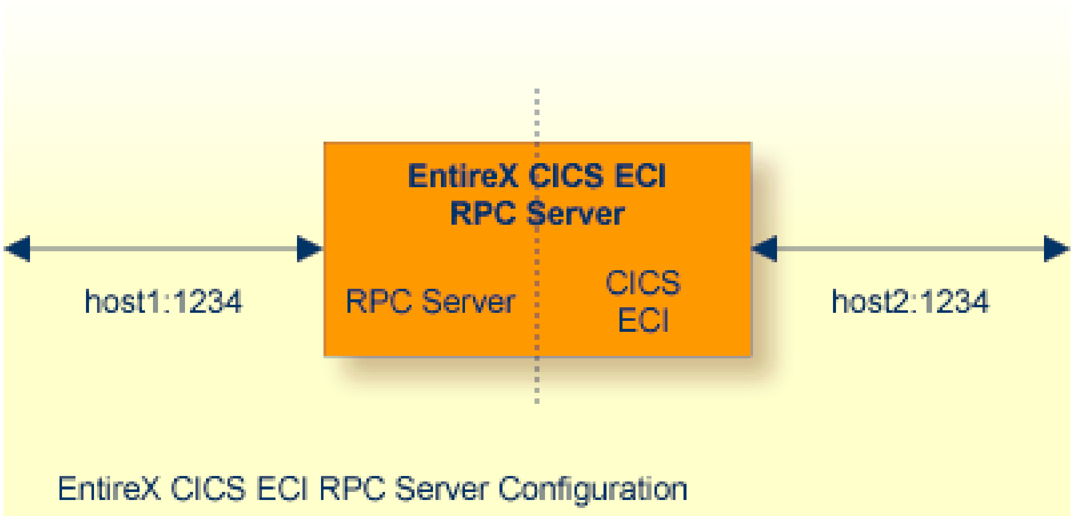
The CICS ECI RPC Server is contained in the file *entirex.jar*.

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
host1:1234
RPC Server
CICS ECI
host2:1234
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 send 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)
			`BEST_COMPRESSION` `9`
			`BEST_SPEED` `1`
			`DEFAULT_COMPRESSION` `-1`, mapped
			`DEFLATED` `8`
			`NO_COMPRESSION` `0`
			`N` `0`
			`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 Advan* *Applications - EntireX Java ACI*.
`entirex.server.verbose`	`-verbose`	`no`	Verbose output to standard output ye
`entirex.server.logfile`	`-logfile`		Path and name of the trace output file
`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 Managen 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. **Note:** Enable conversion in the Broker attribute file so the data can be converted correctly, typically by setting service-specific attribute `CONVERSION` to "`SAGTCHA`". 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 server expects SVM files (extension `.svm`). SVM files are generated by the IDL Extractor for COBOL and COBOL Wrapper. See *Server Mapping Deployment*.

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 Handling SVM Files

A server mapping file (SVM) enables the RPC server to correctly support special COBOL syntax such as `REDEFINE`s, `JUSTIFIED`, `SYNCHRONIZE` and `OCCURS DEPENDING ON` clauses, `LEVEL-88` fields, etc. If one of these elements is used, the EntireX Workbench automatically extracts an SVM file in addition to the IDL (interface definition language), or an SVM file is generated by the COBOL Wrapper for a server skeleton. The SVM file is used at runtime to marshal and unmarshal the RPC data stream.

SVM Files in the EntireX Workbench

In the *EntireX Workbench*, an SVM file has to relate to an appropriate IDL file. Therefore, you always have to keep the IDL file and the SVM file together in the same folder.

If there is an SVM file and a corresponding IDL file,

- at least one of the IDL programs in the corresponding IDL file requires server-mapping information to correctly call the target server. For those IDL programs, there is an SVM entry (line) in the Workbench SVM file.
- deployment of the SVM file to the RPC server is mandatory, see *Server Mapping Deployment*.

If there is an IDL file but no corresponding SVM file,

- there is no IDL program that requires server mapping information.

SVM Files in the RPC Server

CICS ECI RPC Server

For a CICS ECI RPC Server, copy the SVM files of the Workbench manually to a directory (folder) as operating system files. This directory (folder) is provided to the CICS ECI RPC Server with the property `cics.mapping.folder`. See ***Configuring the CICS ECI Side***.

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

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

See also *Deploying a Server Mapping File* in the COBOL Wrapper documentation.

webMethods EntireX Adapter

For webMethods EntireX Adapter for Integration Server, do not change the location of the generated SVM file. It has to be kept in the same folder as the IDL file and will be picked up automatically together with the IDL file when an adapter connection for CICS ECI is generated. For more information, see the EntireX Adapter documentation under *webMethods > Mainframe Integration* on the **Software AG Product Documentation** website.

Source Control of SVM Files

Because SVM entries within an SVM file contain text data only, a Workbench SVM file is text-based (although it is not intended for human consumption). Therefore, you can include it in your source control management together with the IDL file and the COBOL source(s) as a triplet that should always be kept in sync.

Change Management of SVM Files

For a CICS ECI RPC Server, change management of the SVM directory (see ***SVM Files in the RPC Server***) is similar to change management within ordinary operating system directories (folders). All updates to the SVM directory done after a backup must be kept.

All Workbench SVM files added since the last backup should be available.

Compare SVM Files

For SVM files in the *EntireX Workbench* format, you can use a third party file/text compare tool to check if two files are identical.

The SVM entries (corresponding to lines in a Workbench SVM file) contain a creation timestamp at offset 276 (decimal) in the format *`YYYYMMDDHHIISST`*. The precision is 1/10 of a second.

List Deployed SVM Files

To list the contents of an SVM directory (see ***SVM Files in the RPC Server***), use the Windows Explorer (for Windows) or the `ls` command (for UNIX).

Check if an SVM File Revision has been Deployed

SVM entries (corresponding to lines in Workbench SVM 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 SVM files stored as operating system files in SVM directories.

When is an SVM File Required?

For the IDL Extractor for COBOL

Interface Type	COBOL Syntax	COBOL Mapping Editor	SVM Required	More Information
CICS with DFHCOMMAREA Calling Convention and IN different to OUT	all		yes	*CICS with DFHCOMMAREA Calling Convention* under *Introduction to the IDL Extractor for COBOL* | *CICS DFHCOMMAREA* under *COBOL Parameter Selection*
CICS Channel Container Calling Convention	all		yes	*CICS with Channel Container Calling Convention*
CICS with DFHCOMMAREA Large Buffer Interface	all		yes	*CICS with DFHCOMMAREA Large Buffer Interface*
IMS MPP Message Interface (IMS Connect)	all		yes	*IMS MPP Message Interface (IMS Connect)*
IMS BMP with Standard Linkage Calling Convention	all		yes	*IMS BMP with Standard Linkage Calling Convention*
Micro Focus with Standard Linkage Calling Convention	BINARY clause		yes	*Micro Focus with Standard Linkage Calling Convention*
all	OCCURS DEPENDING ON clause		yes	*Tables with Variable Size - DEPENDING ON Clause* under *COBOL to IDL Mapping* in the IDL Extractor for COBOL documentation
all	REDEFINES clause		yes	*REDEFINE Clause*

Interface Type	COBOL Syntax	COBOL Mapping Editor	SVM Required	More Information
all	`TRAILING [SEPARATE]` clause		yes	*`SIGN LEADING` and `TRAILING SEPARATE` Clause*
all	`LEADING [SEPARATE]` clause		yes	*`SIGN LEADING` and `TRAILING SEPARATE` Clause*
all	`ALIGNED RIGHT` attribute		yes	
all	all	Rename of program	yes	*The Software AG IDL Tree Pane* under *Mapping Editor User Interface* in the IDL Extractor for COBOL documentation
all	all	Map to operation	yes	*Context Menu* under *The COBOL Parameters Pane*
all	all	Map to constant	yes	*Context Menu*
all	all	Suppress	yes	*Context Menu*
other combinations			no	

For the COBOL Wrapper

This depends on the interface type chosen and the IDL type:

Interface Type	IDL Type	COBOL Wrapper	SVM Required	More Information
CICS with DFHCOMMAREA Large Buffer Interface	all		yes	*CICS with DFHCOMMAREA Large Buffer Interface* under *COBOL Server Interface Types*
CICS with Channel Container Calling Convention	all		yes	*CICS with Channel Container Calling Convention*
IMS BMP with Standard Linkage Calling Convention	all		yes	*IMS BMP with Standard Linkage Calling Convention*
Micro Focus	I2 or I4		yes	*Micro Focus with Standard Linkage Calling Convention* | *IDL Data Types* under *Software AG IDL File* in the IDL Editor documentation
all	IDL unbounded array		yes	*`array-definition`* under *Software AG IDL Grammar* in the IDL Editor documentation

Interface Type	IDL Type	COBOL Wrapper	SVM Required	More Information
all	IDL unbounded group		yes	`group-parameter-definition` under *Software AG IDL Grammar* in the *IDL Editor* documentation
all	all	IDL program name is not a valid COBOL name and is therefore adapted, or the COBOL program name is customized	yes	*Customize Automatically Generated Server Names*
other combinations			no	

Is There a Way to Smoothly Introduce SVM Files?

All EntireX RPC servers can be executed without SVM files. There is no need to install the SVM container (see ***SVM Files in the RPC Server***) as long as you do not use features that require SVM files (see ***When is an SVM File Required?***). You can also call COBOL servers generated or extracted with previous versions of EntireX mixed with a COBOL server that requires SVM files. 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 of the extraction, also an SVM file.
2. Build an EntireX RPC client using any EntireX wrapper. See *EntireX Wrappers*. 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*

See *Basic RPC Server Examples* - `CALC`, `SQUARE` and *Reliable RPC Server Example* - `SENDMAIL`.

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 of the extraction, also an SVM file. 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. See *EntireX Wrappers*. 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*

See *Basic RPC Server Examples* - `CALC`, `SQUARE` and *Reliable RPC Server Example* - `SENDMAIL`.