

webMethods EntireX

EntireX RPC Server for IBM® MQ

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-MQRPC-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 IBM MQ	5
Overview	6
Sending a Message to an IBM MQ Queue	7
Receiving a Message from an IBM MQ Queue	8
Administration using Command Central	8
Worker Models	10
3 Mapping RPC Data to the MQ Message Buffer	11
Mapping IDL as XML	12
Mapping IDL as Text	12
4 Administering the RPC Server for IBM MQ using the Command Central Command Line	15
Creating an RPC Server Instance	16
Configuring an RPC Server Instance	18
Displaying the EntireX Inventory	33
Viewing the Runtime Status	35
Starting an RPC Server Instance	36
Stopping an RPC Server Instance	36
Inspecting the Log Files	37
Changing the Trace Level Temporarily	39
Deleting an RPC Server Instance	40
5 Administering the RPC Server for IBM MQ using the Command Central GUI	43
Logging in to Command Central	44
Creating an RPC Server Instance	45
Configuring an RPC Server Instance	50
Viewing the Runtime Status	56
Starting an RPC Server Instance	57
Stopping an RPC Server Instance	59
Inspecting the Log Files	61
Changing the Trace Level Temporarily	62
Deleting an RPC Server Instance	62
6 Administering the EntireX RPC Server for IBM® MQ	65
Customizing the RPC Server	66
Configuring the RPC Server Side	68
Configuring the IBM MQ Side	71
Starting the RPC Server	73
Stopping the RPC Server	73
Pinging the RPC Server	74
Using SSL/TLS with the RPC Server	74
Running an EntireX RPC Server as a Windows Service	75

Activating Tracing for the RPC Server	76
7 Advanced RPC Server for IBM MQ Functionality	77
Support for Request/Reply Scenarios	78
Handling of Correlation ID	78
Character Encoding Issues	78
User Exit for Message Processing	79
Transactional Behavior	80

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 IBM MQ

- Overview 6
- Sending a Message to an IBM MQ Queue 7
- Receiving a Message from an IBM MQ Queue 8
- Administration using Command Central 8
- Worker Models 10

The EntireX RPC Server for IBM® MQ runs as an RPC server and processes RPC client calls. It is used to send messages to and receive messages from an IBM MQ queue. This means that existing EntireX wrappers can be used for communication with IBM MQ. This chapter covers the following topics:

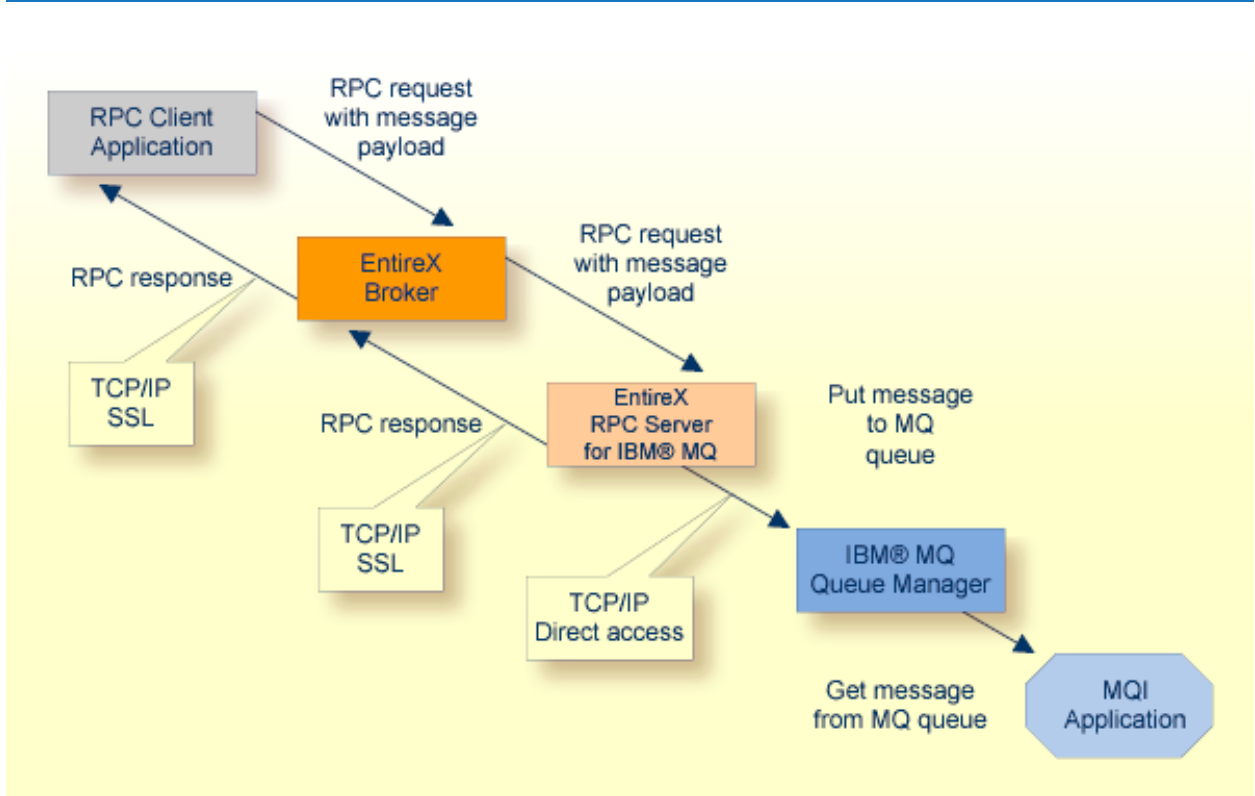
Overview


The EntireX RPC Server for IBM® MQ allows standard RPC clients to send and receive asynchronous and synchronous messages to an IBM® MQ queue manager. The RPC Server for IBM MQ uses the IBM® MQ base Java classes from IBM. It can connect to an IBM® MQ either as an IBM® MQ client using TCP/IP (*client mode*) or in so-called *bindings mode* where it is connected directly to IBM® MQ running on the same machine. Note that on z/OS, only bindings mode is supported.

The RPC Server for IBM MQ runs as an RPC server and processes RPC client calls. An RPC client can send an asynchronous message (MQPUT call) if it uses a program with IN parameters. An RPC client can receive an asynchronous message if it uses a program with OUT parameters (MQGET call). The receiver must use the same parameters in a program as the sender, but with the direction OUT instead of IN. Processing of synchronous messages (request/reply scenario) is possible if the program uses a mixture of IN and OUT parameters. For possibilities on how RPC data is mapped to MQ messages, see [Mapping RPC Data to the MQ Message Buffer](#). The images below illustrate message transport when sending and receiving messages. If the RPC client application uses conversational RPC, the MQ calls are issued transactionally (using the SYNCPOINT option), a Backout Conversation will send a backout to the queue manager, and a Commit Conversation will send a commit to the queue manager.

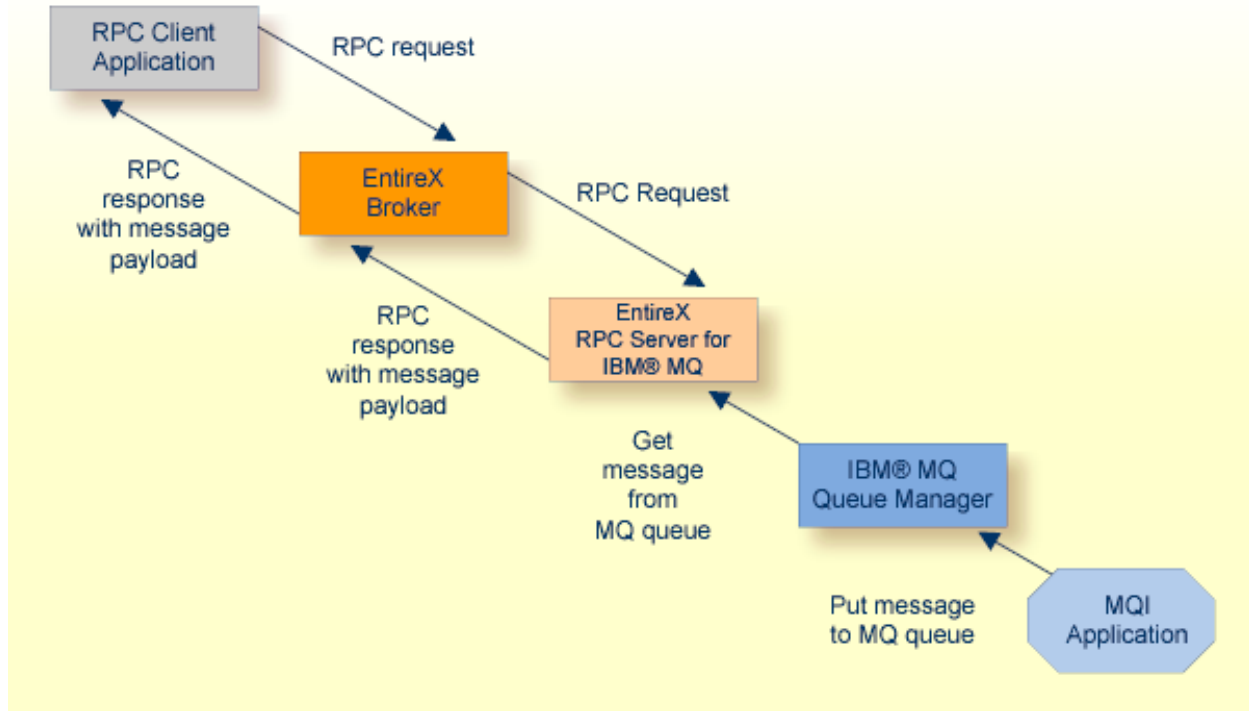
The RPC Server for IBM MQ registers to one RPC service. On the MQ side it uses one input queue and one output queue.

Sending a Message to an IBM MQ Queue



-  **Note:** All messages sent to an RPC Server for IBM MQ instance via a specific RPC service are put on the same MQ output queue.

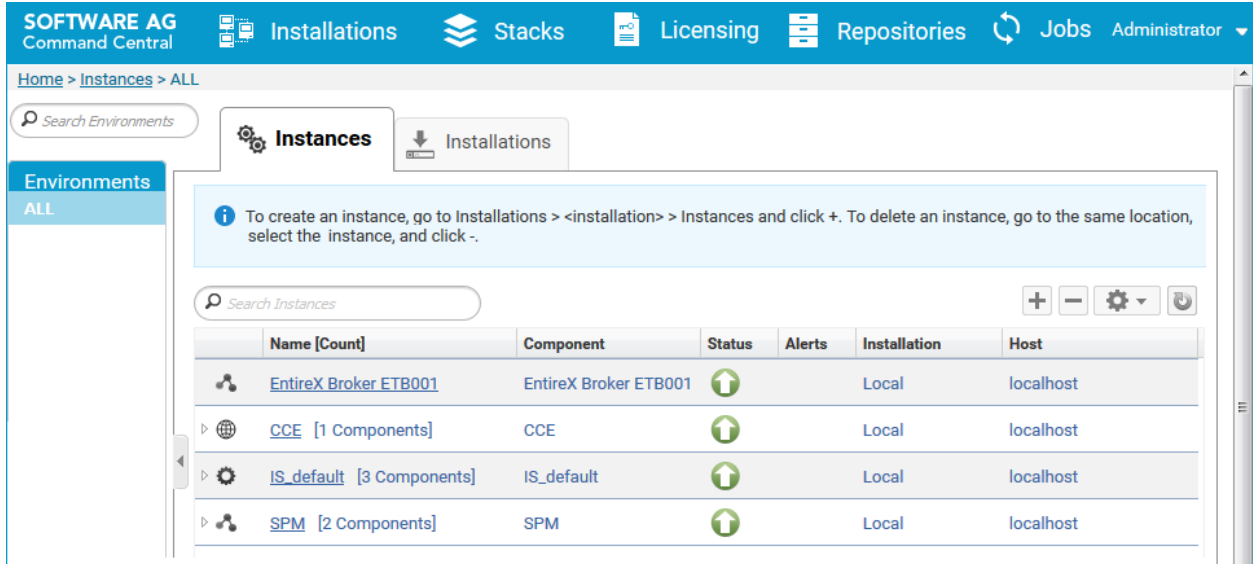
Receiving a Message from an IBM MQ Queue



Note: All messages retrieved by the RPC Server for IBM MQ from the MQ input queue are passed to the same RPC service. Messages are retrieved in the order they appear on the queue.

Administration using Command Central

Software AG Command Central is a tool that enables you to manage your Software AG products remotely from one location. Command Central offers a browser-based user interface, but you can also automate tasks by using commands to remotely execute actions from a terminal or custom script (for example CI servers such as Jenkins, or generic configuration management tools such as Puppet or Chef).



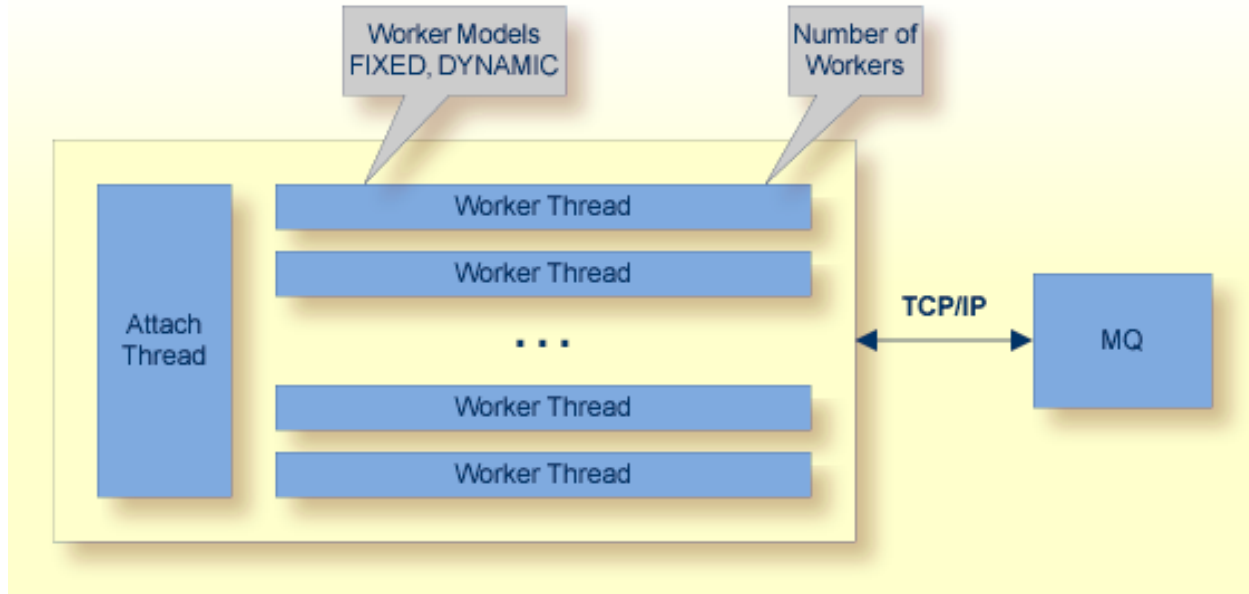
Command Central can assist with the following configuration, management, and monitoring tasks:

- Infrastructure engineers can see at a glance which products and fixes are installed, where they are installed, and compare installations to find discrepancies.
- System administrators can configure environments by using a single web user interface or command-line tool. Maintenance involves minimum effort and risk.
- Release managers can prepare and deploy changes to multiple servers using command-line scripting for simpler, safer lifecycle management.
- Operators can monitor server status and health, as well as start and stop servers from a single location. They can also configure alerts to be sent to them in case of unplanned outages.

The Command Central graphical user interface is described under [Administering the RPC Server for IBM MQ using the Command Central GUI](#). For the command-line interface, see [Administering the RPC Server for IBM MQ using the Command Central Command Line](#).

The core Command Central documentation is provided separately and is also available under **Guides for Tools Shared by Software AG Products** on the Software AG documentation website.

Worker Models



RPC requests are worked off inside the RPC server in worker threads. 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 for IBM MQ can adjust the number of worker threads to the number of parallel requests. 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 with the Command Central GUI, see [Worker Scalability](#) under *Configuration > Server*.

For technical details, see property `entirex.server.fixedservers` under *Administering the RPC Server for IBM MQ with Local Scripts*.

3 Mapping RPC Data to the MQ Message Buffer

- Mapping IDL as XML 12
- Mapping IDL as Text 12

Mapping IDL as XML

IDL is mapped to XML format using the *XML Mapping Editor*. The outcome of this process is an XML mapping file (Designer file with extension xmm). This resulting XMM file has to be specified by the configuration property `entirex.bridge.xmm`.

Mapping IDL as Text

The RPC Server for IBM MQ uses the predefined mapping of IDL data types to the MQ message buffer. See below. If your programs use arrays of groups, set the property `entirex.bridge.marshalling` to "Natural" or "COBOL". If your programs do *not* use arrays of groups, do not set `entirex.bridge.marshalling`.

Data Type	Description	Format	Note
<i>Anumber</i>	Alphanumeric	<i>number</i> bytes, encoding the characters.	
AV	Alphanumeric variable length	Bytes up to the end of the buffer.	1, 4
<i>AVnumber</i>	Alphanumeric variable length with maximum length	Bytes up to the end of the buffer, maximum length <i>number</i> .	1
<i>Bnumber</i>	Binary	byte[]	5
BV	Binary variable length		5
<i>BVnumber</i>	Binary variable length with maximum length		5
<i>Knumber</i>	Kanji	Same as data type A.	
KV	Kanji variable length	Same as data type AV.	1, 4
<i>KVnumber</i>	Kanji variable length with maximum length	Same as data type <i>AVnumber</i> .	1
F4	Floating point (small)		5
F8	Floating point (large)		5
I1	Integer (small)	<i>sign</i> (+, -) and 3 bytes (digits).	
I2	Integer (medium)	<i>sign</i> (+, -) and 5 bytes (digits).	
I4	Integer (large)	<i>sign</i> (+, -) and 10 bytes (digits).	
<i>Nnumber1</i> [. <i>number2</i>]	Unpacked decimal	<i>sign</i> (+, -), <i>number1</i> bytes (digits) [<i>number2</i>] bytes (digits), no decimal point.	
<i>NUnumber1</i> [. <i>number2</i>]	Unpacked decimal unsigned		5
<i>Pnumber1</i> [. <i>number2</i>]	Packed decimal	<i>sign</i> (+, -), <i>number1</i> bytes (digits) [<i>number2</i>] bytes (digits), no decimal point.	

Data Type	Description	Format	Note
<i>PUnumber1[.number2]</i>	Packed decimal unsigned		5
L	Logical	1 byte: X for true, all other false.	
D	Date	YYYYMMDD.	2
T	Time	YYYYMMDDhhmmss.S.	3
<i>Unumber</i>	Unicode		5
UV	Unicode variable length		5
<i>UVnumber</i>	Unicode variable length with maximum length		5

See also the hints and restrictions valid for all languages under *IDL Data Types* in the IDL Editor documentation.



Notes:

1. Only as last value.
2. *YYYY* year, *MM* month, *DD* day.
3. *YYYY* year, *MM* month, *DD* day, *hh* hour, *mm* minute, *ss* second, *S* tenth of a second.
4. Not possible when using COBOL.
5. Data type not supported.

4 Administering the RPC Server for IBM MQ using the Command Central Command Line

- Creating an RPC Server Instance 16
- Configuring an RPC Server Instance 18
- Displaying the EntireX Inventory 33
- Viewing the Runtime Status 35
- Starting an RPC Server Instance 36
- Stopping an RPC Server Instance 36
- Inspecting the Log Files 37
- Changing the Trace Level Temporarily 39
- Deleting an RPC Server Instance 40

This chapter describes how to administer the EntireX RPC Server for IBM® MQ, using the Command Central command-line interface.

Administering the RPC Server for IBM MQ using the Command Central GUI is described under [Administering the RPC Server for IBM MQ using the Command Central GUI](#). The core Command Central documentation is provided separately and is also available under **Guides for Tools Shared by Software AG Products** on the Software AG documentation website.

Creating an RPC Server Instance

The following table lists the parameters to include when creating an EntireX RPC instance, using the Command Central `create instances` commands.

Command	Parameter	Value	Description
sagcc create instances	<i>node_alias</i>	<i>name</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>type</i>	RpcServerIbmMq	Required. EntireXCore instance type of RPC server. Must be "RpcServerIbmMq".
	<i>product</i>	EntireXCore	Required. Must be set to "EntireXCore".
	<i>instance.name</i>	<i>name</i>	Required. Name of the runtime component, for example "MyRpcServer".
	<i>install.service</i>	true <u>false</u>	Optional. Register Windows Service for automatic startup. Default is false. If this parameter is true, the RPC server can be controlled by the Windows Service Control Manager.
	<i>server.address</i>	<i>class/server/service</i>	Required. The case-sensitive RPC server address has the format: CLASS/SERVER/SERVICE.
	<i>server.adminport</i>	1025-65535	Required. The administration port in range from 1025 to 65535.
	<i>broker.transport</i>	ssl <u>tcp</u>	Transport over TCP or SSL. Default is TCP.
	<i>broker.host</i>	<i>name</i>	Required. EntireX Broker host name or IP address.
	<i>broker.port</i>	1025-65535	Required. Port number in range from 1025 to 65535.
	<i>broker.user</i>	<i>user</i>	Optional. The user ID for secured access to the broker.
	<i>broker.password</i>	<i>password</i>	Optional. The password for secured access to the broker.

Command	Parameter	Value	Description
	mq.bindingsmode	yes no	Required. "yes"=bindings mode, "no"=client mode.
	mq.host	<i>name</i>	Required for client mode only. Host name of IP address of the MQ server.
	mq.port	<i>n</i>	Required for client mode only. Port of the MQ server.
	mq.channel	<i>name</i>	Required for client mode only. Channel name used to connect to the MQ server.
	mq.queuemanager	<i>name</i>	Optional. Name of the queue manager. If not specified, a connection is made to the default queue manager.
	mq.inputqueue	<i>name</i>	Optional. Name of input queue that is used for MQ GET operations.
	mq.outputqueue	<i>name</i>	Optional. Name of output queue that is used for MQ PUT operations.
	mq.javainstallpath	<i>path</i>	Required. MQ Java installation path. If set, take the value of the environment variable MQ_JAVA_INSTALL_PATH from the MQ installation.
	mq.javalibpath	<i>path</i>	Required for bindings mode only. MQ Java library path. If set, take the value of the environment variable MQ_JAVA_LIB_PATH from the MQ installation.
	mq.user	<i>userid</i>	Required if MQ server is running with security.
	mq.password	<i>password</i>	Required if MQ server is running with security.

Example

- To create a new instance for an installed EntireX of the type "RpcServerIbmMq", with name "MyRpcServer", with server address "RPC/SRV1/CALLNAT", using administration port 5757, with broker host name "localhost", listening on broker port 1971, with bindings mode="yes", in the installation with alias name "local":

```
sagcc create instances local EntireXCore type=RpcServerIbmMq
instance.name=MyRpcServer server.address=RPC/SRV1/CALLNAT server.adminport=5757
broker.host=localhost broker.port=1971 mq.bindingsmode=yes
mq.javainstallpath=myMQJavaInstallPath
```

Information about the creation job - including the job ID - is displayed.

Configuring an RPC Server Instance

Here you can administer the parameters of the RPC Server for IBM MQ. Any changes to parameters will be used the next time you start the RPC server.

- [Broker](#)
- [Configuration File](#)
- [MQ](#)
- [Monitoring KPIs](#)
- [Server](#)
- [Trace Level](#)

Broker

Here you can administer the parameters used for communication between the RPC Server for IBM MQ and EntireX Broker.

- [Parameters](#)
- [Displaying the Broker Settings of the RPC Server](#)
- [Updating the Broker Settings of the RPC Server](#)

Parameters

Parameter	Value	Description
BrokerTransport	TCP SSL	Transport over TCP or SSL. Default is TCP.
BrokerHost	<i>name</i>	Required. EntireX Broker host name or IP address.
BrokerPort	1025-65535	Required. Port number in range from 1025 to 65535.
BrokerUser	<i>user</i>	Optional. The user ID for secured access to the broker.
BrokerPassword	<i>password</i>	Optional. The password for secured access to the broker.
BrokerEncoding	<i>codepage</i>	Required. Encoding used for the communication between the RPC server and EntireX Broker.
BrokerSslTrustStore	<i>filename</i>	Optional. Specifies the location of SSL trust store.
BrokerSslVerifyServer	true false	Optional. The RPC server as SSL client checks the identity of the broker as SSL server.

Displaying the Broker Settings of the RPC Server

Command	Parameter	Description
sagcc get configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "BROKER".
	<i>-o file</i>	Optional. Specifies the file where you want the output written.

Example 1

- To display the Broker parameters of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local":

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer BROKER
```

Example 2

- To store the Broker parameters in the file *broker.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer BROKER
-o broker.json
```

Resulting output file in JSON format:

```
{
  "BrokerHost": "localhost",
  "BrokerPort": "1971",
  "BrokerTransport": "TCP",
  "BrokerUser": "testuser",
  "BrokerPassword": "",
  "BrokerEncoding": "Cp1252",
  "BrokerSslTrustStore": "",
  "BrokerSslVerifyServer": "true"
}
```

Updating the Broker Settings of the RPC Server

Command	Parameter	Description
sagcc update configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "BROKER".
	<i>-i file</i>	Optional. Specifies the file from where you want the input read.

Example

- To load the Broker parameters of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" from the file *broker.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer
BROKER -i broker.json
```

See [Example 2](#) above for sample input file.

Configuration File

Here you can administer the configuration file of the RPC Server for IBM MQ. Any changes will take effect after the next restart.

- [Displaying the Content of the RPC Server Configuration File](#)
- [Updating the Content of the RPC Server Configuration File](#)

Displaying the Content of the RPC Server Configuration File

Command	Parameter	Description
sagcc get configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "CONFIGURATION".
	<i>-o file</i>	Optional. Specifies the file where you want the output written.

Example 1

- To display the configuration file of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local":

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer
CONFIGURATION
```

Example 2

- To store the contents of the configuration file in the text file *configuration.txt* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer
CONFIGURATION -o configuration.txt
```

Updating the Content of the RPC Server Configuration File

Command	Parameter	Description
sagcc update configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "CONFIGURATION".
	<i>-i file</i>	Optional. Specifies the file from where you want the input read.

Example

- To load the contents of configuration file *configuration.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer  
CONFIGURATION -i configuration.json
```

MQ

Here you can modify the MQ-specific parameters. This section covers the following topics:

- [Parameters](#)
- [Displaying the MQ-specific Parameters](#)
- [Updating the MQ-specific Parameters](#)

Parameters

Parameter	Description
IbmMqBindingsMode	Required. "yes"=bindings mode, "no"=client mode.
IbmMqHost	Required for client mode only. Host name or IP address of the MQ server.
IbmMqPort	Required for client mode only. Port of the MQ server.
IbmMqChannel	Required for client mode only. Channel name used to connect to the MQ server.
IbmMqQueueManager	Optional. Name of the queue manager. If not specified, a connection is made to the default queue manager.
IbmMqInputQueue	Optional. Name of input queue that is used for MQ GET operations.
IbmMqOutputQueue	Optional. Name of output queue that is used for MQ PUT operations.
IbmMqJavaInstallPath	Required. MQ Java installation path. If set, take the value of the environment variable MQ_JAVA_INSTALL_PATH from the MQ installation.
IbmMqJavaLibPath	Required for bindings mode only. MQ Java library path. If set, take the value of the environment variable MQ_JAVA_LIB_PATH from the MQ installation.
IbmMqCharacterSetId	Coded Character Set Identification used by the RPC Server for IBM MQ (which acts as an MQ client). Not used in bindings mode. (Numeric value).
IbmMqWaitTime	Wait interval for MQGET call in milliseconds.
IbmMqPriority	Message priority for messages sent to MQ.
IbmMqSsl	Configuration for SSL connection to MQ server. Not used in bindings mode.
IbmMqMapping	Name of XMM (XML Mapping) file if MQ message payload is XML/SOAP. If this is specified, messages to/from MQ will be converted to XML/SOAP.
IbmMqUser	Required if MQ server is running with security.
IbmMqPassword	Required if MQ server is running with security.
IbmMqUserExit	Class name for MQ Server user exit.
IbmMqUserExitClasspath	URL of the classpath for user exit.

Displaying the MQ-specific Parameters

Command	Parameter	Description
sagcc get configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "IBM-MQ".
	<i>-o file</i>	Optional. Specifies the file where you want the output written.

Example 1

- To display the MQ-specific parameters of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local":

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer IBM-MQ
```

Example 2

- To store the MQ-specific parameters in the file *mq.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer IBM-MQ -o mq.json
```

Resulting output file in JSON format:

```
{ "IbmMqBindingsMode": "yes",
  "IbmMqHost": "localhost",
  "IbmMqPort": "1414",
  "IbmMqChannel": "MQ_CHANNEL",
  "IbmMqQueueManager": "MQ_QUEUEMANAGER",
  "IbmMqInputQueue": "Q2",
  "IbmMqOutputQueue": "Q1",
  "IbmMqJavaInstallPath": "myMQJavaInstallPath"
  "IBMMqJavaLibPath": "",
  "IbmMqUser": "",
  "IbmMqMapping": "example.xmm",
  "IbmMqCharacterSetId": "",
  "IbmMqMarshalling": "",
  "IbmMqUserExit": "",
  "IbmMqUserExitClasspath": "",
  "IbmMqWaitTime": "10000",
  "IbmMqSsl": "",
  "IbmMqPriority": "" }
```

Updating the MQ-specific Parameters

Command	Parameter	Description
sagcc update configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "IBM-MQ".
	<i>-i file</i>	Optional. Specifies the file from where you want the input read.

Example

- To load the MQ-specific parameters of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" from the file *mq.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer IBM-MQ
-i mq.json
```

See [Example 2](#) above for sample output file.

Monitoring KPIs

Here you can administer margins of monitored key performance indicators (KPIs) available for the RPC Server for IBM MQ: Active Workers and Busy Workers.

- [Parameters](#)
- [Displaying the Monitoring KPIs](#)
- [Updating the Monitoring KPIs](#)

Parameters

Key performance indicators (KPIs) enable you to monitor the health of your RPC Server for IBM MQ. The following KPIs help you administer, troubleshoot, and resolve performance issues:

KPI	Setting
Absolute number of Active Workers	entirex.generic.kpi.1.max=20
Critical alert relative to maximum	entirex.generic.kpi.1.critical=0.95
Marginal alert relative to maximum	entirex.generic.kpi.1.marginal=0.80
Absolute number of Busy Workers	entirex.generic.kpi.2.max=20
Critical alert relative to maximum	entirex.generic.kpi.2.critical=0.95
Marginal alert relative to maximum	entirex.generic.kpi.2.marginal=0.80

Do not change the other properties!

Displaying the Monitoring KPIs

Command	Parameter	Description
sagcc get configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "EXX-MONITORING-KPIS".
	<i>-o file</i>	Optional. Specifies the file where you want the output written.

Example 1

- To display the monitoring KPI properties of RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer
MONITORING-KPI
```

Example 2

- To store the monitoring KPI properties in the file *my.properties* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer
MONITORING-KPI -o my.properties
```

Resulting output file in text format:

```
entirex.entirex.spm.version=10.5.0.0.473
entirex.generic.kpi.1.critical=0.95
entirex.generic.kpi.1.id=\#1
entirex.generic.kpi.1.marginal=0.80
entirex.generic.kpi.1.max=20
entirex.generic.kpi.1.name=Active Workers
entirex.generic.kpi.1.unit=
entirex.generic.kpi.1.value=0
entirex.generic.kpi.2.critical=0.95
entirex.generic.kpi.2.id=\#2
entirex.generic.kpi.2.marginal=0.80
entirex.generic.kpi.2.max=20
entirex.generic.kpi.2.name=Busy Workers
entirex.generic.kpi.2.unit=
entirex.generic.kpi.2.value=0
```

Updating the Monitoring KPIs

Command	Parameter	Description
sagcc update configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "EXX-MONITORING-KPIS".
	<i>-i file</i>	Optional. Specifies the file from where you want the input read.

Example

- To load the contents of file *my.properties* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer  
MONITORING-KPI -i my.properties
```


Server

Here you can administer the parameters defining the registration name, the administration port and the behavior of the RPC Server for IBM MQ.

- [Parameters](#)
- [Displaying the Server Settings](#)
- [Updating the Server Settings](#)

Parameters

Parameter	Value	Description
ServerAddress	<i>class/server/service</i>	Required. The case-sensitive RPC server address has the format: CLASS/SERVER/SERVICE.
ServerAdminport	1025-65535	Required. The administration port in range from 1025 to 65535.
ReconnectionAttempts	<i>n</i>	Required. Number of reconnection attempts to the broker. When the number of attempts is reached and a connection to the broker is not possible, the RPC Server stops.
WorkerScalability	<i>true</i> <i>false</i>	You can either have a fixed or dynamic number of workers. Default is dynamic (<i>true</i>). For more information see Worker Models .
FixNumber	1-255	Required. Fixed number of workers. Must be a number in range from 1 to 255.
MinWorkers	1-255	Required. Minimum number of workers. Must be a number in range from 1 to 255.
MaxWorkers	1-255	Required. Maximum number of workers. Must be a number in range from 1 to 255.

Displaying the Server Settings

Command	Parameter	Description
sagcc get configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "SERVER".
	<i>-o file</i>	Optional. Specifies the file where you want the output written.

Example 1

- To display the server parameters of RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer SERVER
```

Example 2

- To store the server parameters in the file *server.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer SERVER
-o server.json
```

Resulting output file in JSON format:

```
{
  "ServerAddress": "RPC/SRV1/CALLNAT",
  "ServerAdminport": "4711",
  "ReconnectionAttempts": "15",
  "WorkerScalability": "true",
  "FixNumber": "5",
  "MinWorkers": "1",
  "MaxWorkers": "10"
}
```

Updating the Server Settings

Command	Parameter	Description
sagcc update configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "SERVER".
	<i>-i file</i>	Optional. Specifies the file from where you want the input read.

Example

- To load the server parameters from the file *server.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer  
SERVER -i server.json
```

See [Example 2](#) above for sample input file.

Trace Level

Here you can set the trace level of the RPC Server for IBM MQ.

- [Parameters](#)
- [Displaying the Trace Level](#)
- [Updating the Trace Level](#)

Parameters

Parameter	Value	Description
TraceLevel	0 1 2 3	One of the following levels: 0 - None - No trace output (default). 1 - Standard - Minimal trace output. 2 - Advanced - Detailed trace output. 3 - Support - Support diagnostic. Use only when requested by Software AG support.

Displaying the Trace Level

Command	Parameter	Description
sagcc get configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "TRACE".
	<i>-o file</i>	Optional. Specifies the file where you want the output written.

Example 1

- To display the trace level of RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer TRACE
```

Example 2

- To store the trace level in the file *trace.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer TRACE
-o trace.json
```

Resulting output file in JSON format:

```
{
  "TraceLevel": "0"
}
```

Updating the Trace Level

Command	Parameter	Description
sagcc update configuration data	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	<i>instanceid</i>	Required. Must be "TRACE".
	<i>-i file</i>	Optional. Specifies the file from where you want the input read.

Example

- To load the trace level parameters from the file *trace.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerIbmMq-MyRpcServer
TRACE -i trace.json
```

See [Example 2](#) above for sample input file.

Displaying the EntireX Inventory

Listing all Inventory Components

The following table lists the parameters to include, when listing all EntireX instances, using the Command Central `list inventory` commands.

Command	Parameter	Description
sagcc list inventory components	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>component_id</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".

Example

- To list inventory components of instance EntireX in the installation with alias name "local":

```
sagcc list inventory components local EntireXCore*
```

A list of all EntireX RPC Server runtime components will be displayed.

Viewing the Runtime Status

The following table lists the parameters to include when displaying the state of an EntireX component, using the Command Central `get monitoring` commands.

Command	Parameter	Description
sagcc get monitoring state	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".

Example

- To display state information about the RPC Server for IBM MQ:

```
sagcc get monitoring state local EntireXCore-RpcServerIbmMq-MyRpcServer
```

Runtime status and runtime state will be displayed.

- Runtime *status* indicates whether a runtime component is running or not. Examples of a runtime status are ONLINE or STOPPED.
- Runtime *state* indicates the health of a runtime component by providing key performance indicators (KPIs) for the component. Each KPI provides information about the current use, marginal use, critical use and maximum use.

Starting an RPC Server Instance

The following table lists the parameters to include when starting an EntireX RPC Server for IBM® MQ, using the Command Central `exec lifecycle` commands.

Command	Parameter	Description
sagcc exec lifecycle start	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".

Example

- To start the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local":

```
sagcc exec lifecycle start local EntireXCore-RpcServerIbmMq-MyRpcServer
```

Information about the job - including the job ID - will be displayed.

Stopping an RPC Server Instance

The following table lists the parameters to include when stopping an EntireX RPC Server for IBM® MQ, using the Command Central `exec lifecycle` commands.

Command	Parameter	Description
sagcc exec lifecycle stop	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".

Example

- To stop the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local":


```
sagcc exec lifecycle stop local EntireXCore-RpcServerIbmMq-MyRpcServer
```

Information about the job - including the job ID - will be displayed.

Inspecting the Log Files

Here you can administer the log files of the RPC Server for IBM MQ. The following table lists the parameters to include when displaying or modifying parameters of the RPC server, using the Command Central `list` commands.

- [List all RPC Server Log Files](#)
- [Getting Content from or Downloading RPC Server Log Files](#)

List all RPC Server Log Files

Command	Parameter	Description
sagcc list diagnostics logs	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".

Example

- To list the log files of RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc list diagnostics logs local EntireXCore-RpcServerIbmMq-MyRpcServer
```

Getting Content from or Downloading RPC Server Log Files

Command	Parameter	Description
sagcc get diagnostics logs	<i>node_alias</i>	Required. Specifies the alias name of the installation in which the runtime component is installed.
	<i>componentid</i>	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	full tail head	Optional. Shows full log file content, or only tail or head.
	export -o <i>file</i>	Optional. Creates a zip file of the logs.

Example 1

- To list the tail of the log file content in the current working directory:

```
sagcc get diagnostics logs local EntireXCore-RpcServerIbmMq-MyRpcServer server.log  
tail
```

Example 2

- To create a zip file *myfile.zip* of the logs:

```
sagcc get diagnostics logs local EntireXCore-RpcServerIbmMq-MyRpcServer export  
-o myfile.zip
```

Changing the Trace Level Temporarily

Here you can temporarily change the trace level of a running RPC server. The following table lists the parameters to include when displaying or modifying parameters of an EntireX component, using the Command Central `exec administration` command. The change is effective immediately; there is no need to restart the RPC server.



Note: If you want to set the trace level permanently, see [Trace Level](#) under *Configuring an RPC Server Instance*.

Displaying the Trace Level of a Running RPC Server

Command	Parameter	Description
sagcc exec administration	component	Required. Specifies that a component will be administered.
	node_alias	Required. Specifies the alias name of the installation in which the runtime component is installed.
	Trace	Required. Specifies what is to be administered.
	load tracelevel=?	Required. Get the trace level.
	-f xml json	Required. Specifies XML or JSON as output format.

Example 1

- To display the current trace level of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" in JSON format on stdout:

```
sagcc exec administration component local EntireXCore-RpcServerIbmMq-MyRpcServer
Trace load tracelevel=? -f json
```

Example 2

- To display the current trace level of the RPC Server for IBM MQ "MyRpcServer" in the installation with alias name "local" in XML format on stdout:

```
sagcc exec administration component local EntireXCore-RpcServerIbmMq-MyRpcServer
Trace load tracelevel=? -f xml
```

Updating the Trace Level of a Running RPC Server

Command	Parameter	Description
sagcc exec administration	component	Required. Specifies that a component will be administered.
	node_alias	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".
	Trace	Required. Specifies what is to be administered.
	update tracelevel	Required. Update temporarily the trace level of a running RPC server.
	-f xml json	Required. Specifies XML or JSON as output format.

Example

- To change the current trace level of the running RPC Server with the name "MyRpcServer" in the installation with alias name "local":

```
sagcc exec administration component local EntireXCore-RpcServerIbmMq-MyRpcServer
Trace update tracelevel=2 -f json
```

Deleting an RPC Server Instance

The following table lists the parameters to include when deleting an EntireX RPC Server instance, using the Command Central `delete instances` commands.

Command	Parameter	Description
sagcc delete instances	node_alias	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	Required. The component identifier. The prefix is "EntireXCore-RpcServerIbmMq-".

Example

- To delete an instance of an EntireX RPC Server for IBM® MQ with the name "MyRpcServer" in the installation with alias name "local":

```
sagcc delete instances local EntireXCore-RpcServerIbmMq-MyRpcServer
```

Information about the deletion job - including the job ID - is displayed.

5 Administering the RPC Server for IBM MQ using the Command Central GUI

▪ Logging in to Command Central	44
▪ Creating an RPC Server Instance	45
▪ Configuring an RPC Server Instance	50
▪ Viewing the Runtime Status	56
▪ Starting an RPC Server Instance	57
▪ Stopping an RPC Server Instance	59
▪ Inspecting the Log Files	61
▪ Changing the Trace Level Temporarily	62
▪ Deleting an RPC Server Instance	62

This chapter describes how to administer the EntireX RPC Server for IBM® MQ, using the Command Central graphical user interface.

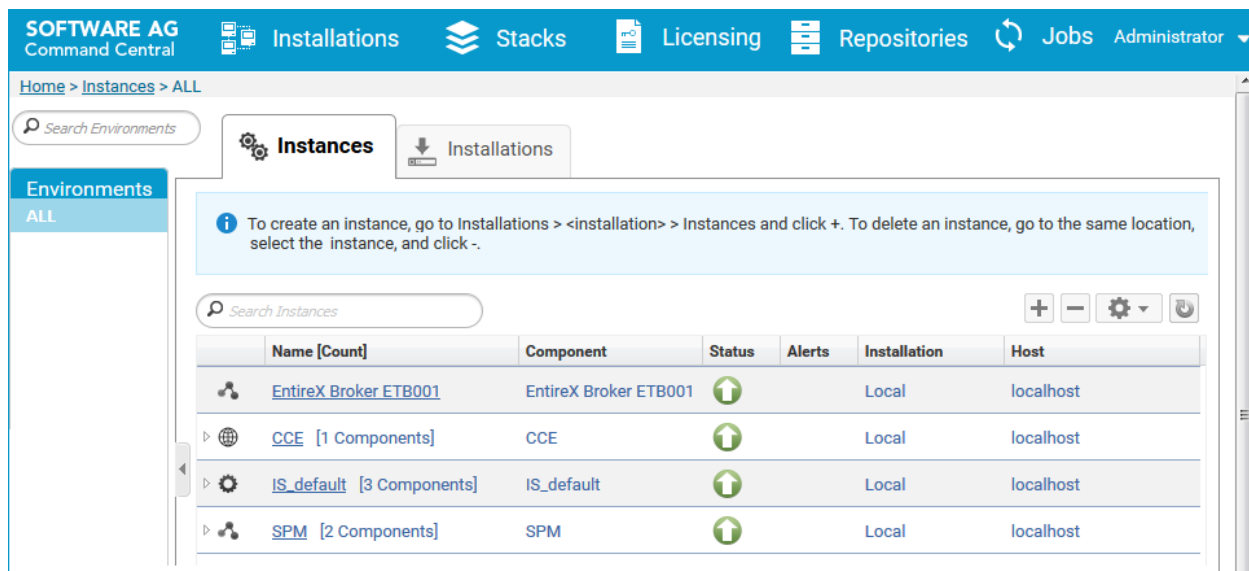
See also *Administering the RPC Server for IBM MQ using the Command Central Command Line*. The core Command Central documentation is provided separately and is also available under **Guides for Tools Shared by Software AG Products** on the Software AG documentation website.

Logging in to Command Central

Open an Internet browser and specify the URL of the Command Central Server as follows: *http://<Command_Central_host>:<Command_Central_port>*. This takes you to the Command Central **Login** page.

On Windows you can also get to the **Login** page from the Command Central Start Menu entry.

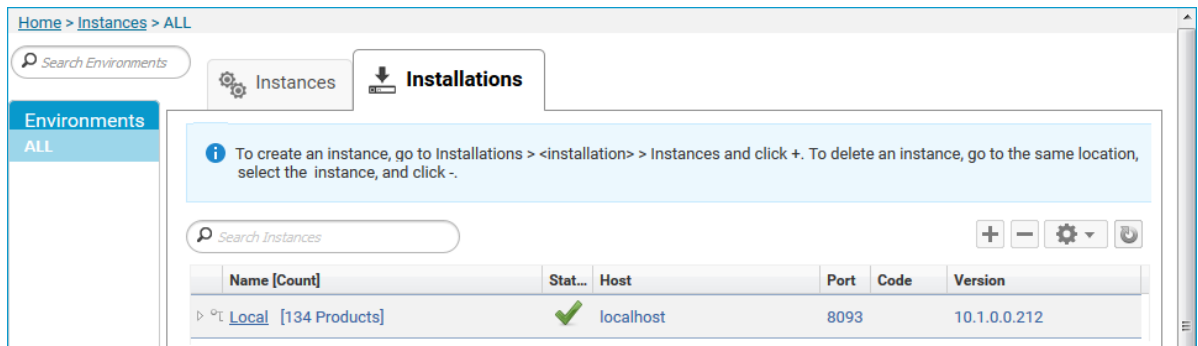
Provide your user credentials in the **Login** page and click **Log In**. This takes you to the page **Home > Instances:**



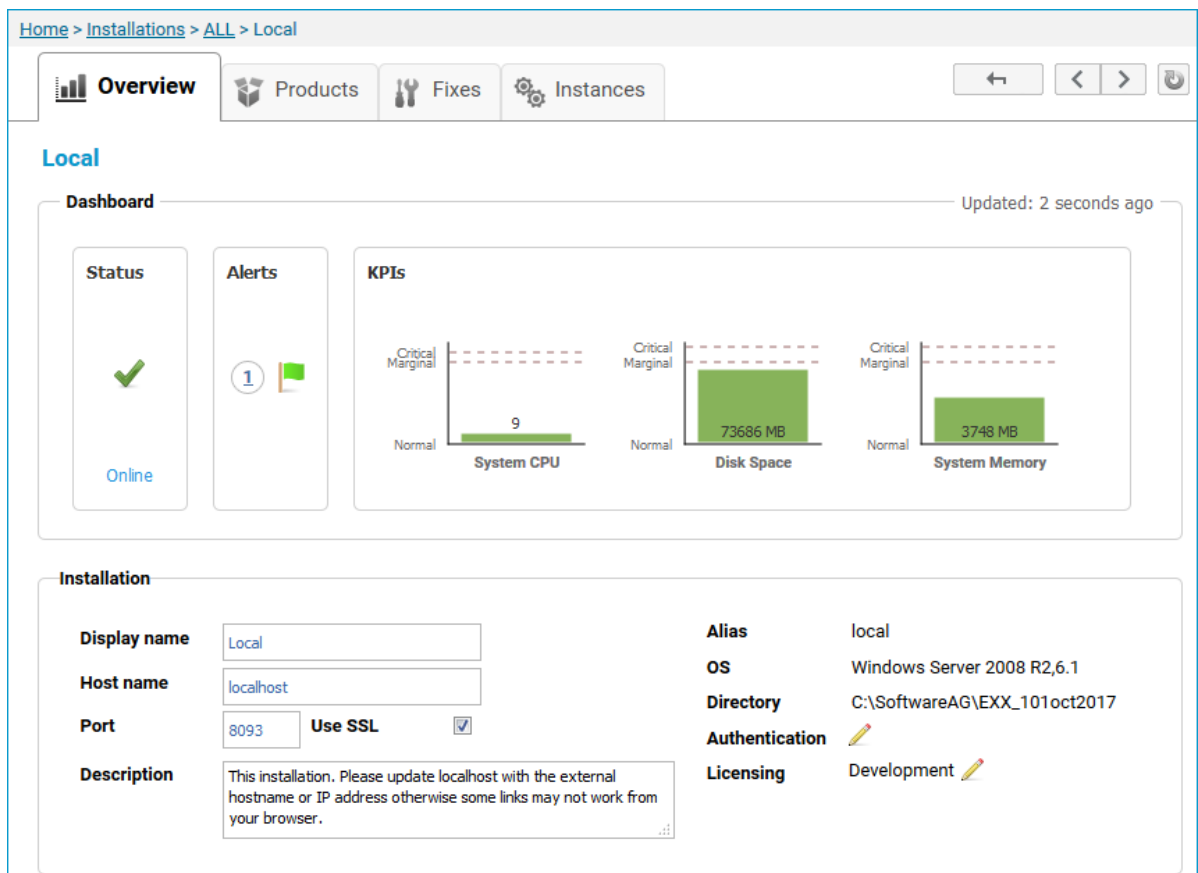
Creating an RPC Server Instance

➤ To create an RPC Server for IBM MQ instance

- 1 In the Command Central home page, click the **Installations** tab.



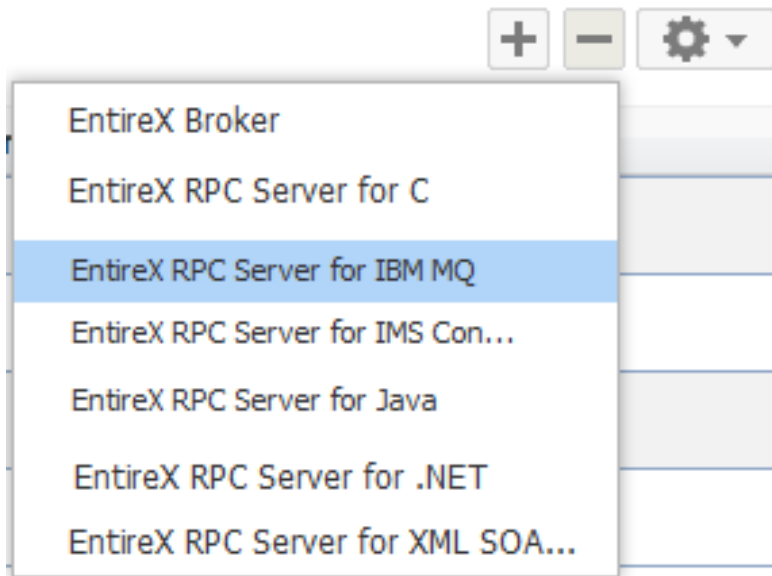
- 2 Click on the desired installation, for example **Local**, where you want to add an RPC Server for IBM MQ instance.



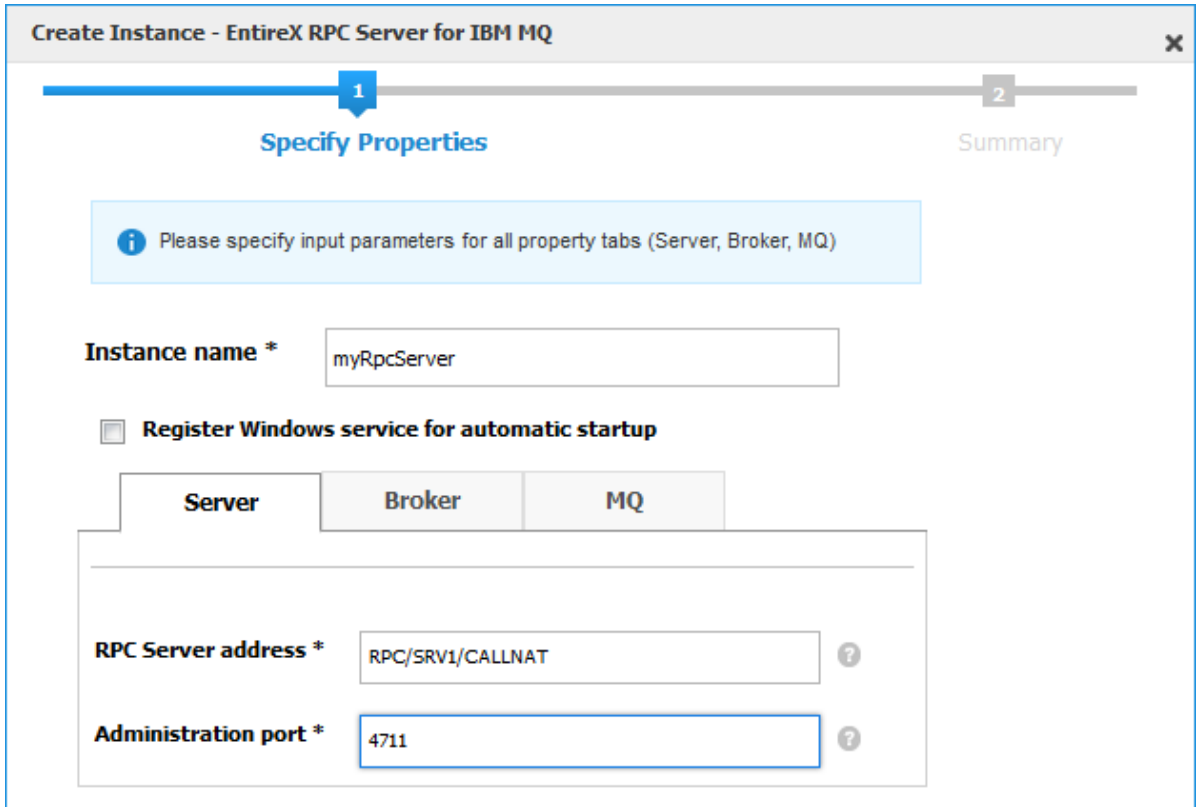
3 Click the **Instances** tab.

	Name [Count]	Component	Status	Alerts	Installation	Host
	EntireX Broker ETB001	EntireX Broker ETB001			Local	localhost
	CCE [1 Components]	CCE			Local	localhost
	IS_default [3 Components]	IS_default			Local	localhost
	SPM [2 Components]	SPM			Local	localhost

4 Click the button in the upper right corner above the list and choose **EntireX RPC Server for IBM® MQ**.



5 In the **Create Instance** wizard, fill in the fields in the main screen and in the **Server, Broker** and **MQ** tabs.



Main Screen

Parameter	Description
Instance name	Required. Name of the runtime component, for example "MyRpcServer".
Register Windows Service for automatic startup	Optional. Register Windows Service for automatic startup. Default is not checked. If this parameter is checked, the RPC server can be controlled by the Windows Service Control Manager.

Server Tab

Parameter	Description
RPC Server address	Required. The case-sensitive RPC server address has the format: CLASS/SERVER/SERVICE.
Administration port	Required. The administration port in range from 1025 to 65535.

Broker Tab

Parameter	Description
Connection	
Transport	Transport over TCP or SSL. Default is TCP.
Broker host	Required. EntireX Broker host name or IP address.
Broker port	Required. Port number in range from 1025 to 65535.
SSL trust store	Optional. Specifies the location of SSL trust store.
Credentials	
User	Optional. The user ID for secured access to the broker.
Password	Optional. The password for secured access to the broker.

MQ Tab

Here you can modify the MQ-specific parameters.

Parameter	Description
Connection	
MQ server	Required. Bindings mode using a direct connection, or client mode using TCP/IP or SSL.
MQ host	Required for client mode only. Host name or IP address of the MQ server.
MQ port	Required for client mode only. Port of the MQ server.
MQ channel	Required for client mode only. Channel name used to connect to the MQ server.
MQ queue manager	Optional. Name of the queue manager. If not specified, a connection is made to the default queue manager.
MQ input queue	Optional. Name of input queue that is used for MQ GET operations.
MQ output queue	Optional. Name of output queue that is used for MQ PUT operations.
MQ Java installation path	Required. MQ Java installation path. If set, take the value of the environment variable MQ_JAVA_INSTALL_PATH from the MQ installation.
MQ Java library path	Required for bindings mode only. MQ Java library path. If set, take the value of the environment variable MQ_JAVA_LIB_PATH from the MQ installation.
Credentials	
MQ user	Required if MQ server is running with security.
MQ password	Required if MQ server is running with security.

- 6 Press **Next** to get to the **Summary** page to verify your input.
- 7 Press **Finish**.

Local

Search Instances

	Name [Count]	Component	Status	Alerts	Installation	Host
	EntireX Broker ETB001	EntireX Broker ETB001	↑		Local	localhost
▶	CCE [1 Components]	CCE	↑		Local	localhost
▶	IS_default [3 Components]	IS_default	↑		Local	localhost
▶	SPM [2 Components]	SPM	↑		Local	localhost

Operation triggered x

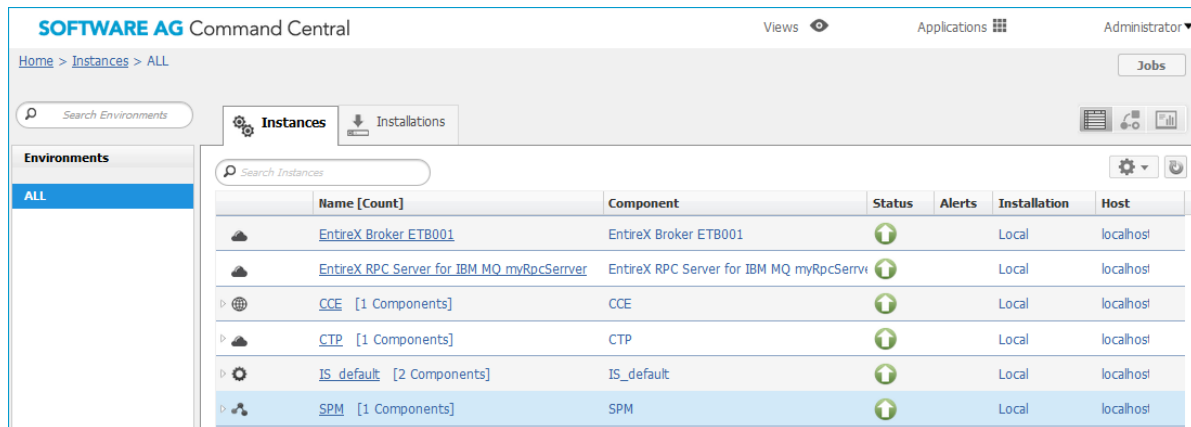
Job operation is started successfully.

The new instance *myRpcServer* appears in the list.

Configuring an RPC Server Instance

➤ To configure an RPC Server for IBM MQ instance

- 1 In the Command Central home page, click the **Instances** tab.



- 2 Click on the link associated with this instance to select the RPC server instance you want to configure.

Overview Configuration Logs Administration

Instance: EntireX RPC Server for IBM MQ myRpcServer

Dashboard

Status
Online

Alerts
1

KPIs

Active Workers: 1
Busy Workers: 0

Details

Display name EntireX RPC Server for IBM MQ my

Component EntireX RPC Server for IBM MQ ...

Host name localhost

Authentication

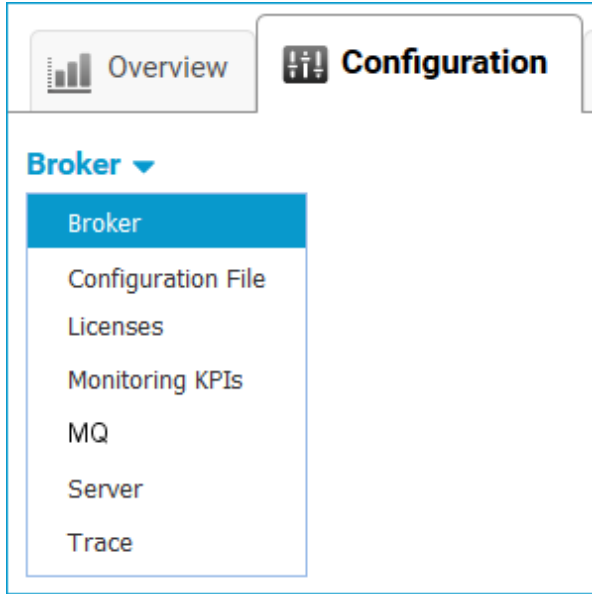
Installation name Local


Installation alias local

Attributes

Name	Value

- 3 Click the **Configuration** tab. EntireX supports the following configuration types, which are presented in a drop-down box when you click the down arrow below the **Configuration** tab label:



 **Note:** All configuration changes require a restart of the instance to take effect.

- [Broker](#)
- [Configuration File](#)
- [Licenses](#)
- [Monitoring KPIs](#)
- [MQ](#)
- [Server](#)
- [Trace Level](#)

Broker

Parameter	Description
Connection	
Transport	Transport over TCP or SSL. Default is TCP.
Broker host	Required. EntireX Broker host name or IP address.
Broker port	Required. Port number in range from 1025 to 65535.
Encoding	Required. Encoding used for the communication between the RPC server and EntireX Broker.
SSL trust store	Optional. Specifies the location of SSL trust store.
SSL verify server	Optional. The RPC server as SSL client checks the identity of the broker as SSL server.
Credentials	
User	Optional. The user ID for secured access to the broker.

Parameter	Description
Password	Optional. The password for secured access to the broker.

Configuration File

Here you can view/edit the configuration file of the RPC Server for IBM MQ.

Licences

Here you can view/set the license file in the EntireX installation. For details see *Point to the License Key for an Instance or Component* under *Working with Standalone Product Installation* in the Command Central documentation.



Note: The license file is used for all EntireX instances in this installation.

Monitoring KPIs

Here you can modify margins of monitored key performance indicators (KPIs) available for the RPC Server for IBM MQ: Active Workers and Busy Workers.

Key performance indicators (KPIs) enable you to monitor the health of your RPC Server for IBM MQ. The following KPIs help you administer, troubleshoot, and resolve performance issues:

KPI	Setting
Absolute number of Active Workers	entirex.generic.kpi.1.max=20
Critical alert relative to maximum	entirex.generic.kpi.1.critical=0.95
Marginal alert relative to maximum	entirex.generic.kpi.1.marginal=0.80
Absolute number of Busy Workers	entirex.generic.kpi.2.max=20
Critical alert relative to maximum	entirex.generic.kpi.2.critical=0.95
Marginal alert relative to maximum	entirex.generic.kpi.2.marginal=0.80

Do not change the other properties!

MQ

Here you can modify the MQ-specific parameters.

Parameter	Description
Connection	
MQ server	Required. Bindings mode using a direct connection, or client mode using TCP/IP or SSL.
MQ host	Required for client mode only. Host name or IP address of the MQ server.
MQ port	Required for client mode only. Port of the MQ server.
MQ channel	Required for client mode only. Channel name used to connect to the MQ server.
MQ queue manager	Optional. Name of the queue manager. If not specified, a connection is made to the default queue manager.
MQ input queue	Optional. Name of input queue that is used for MQ GET operations.
MQ output queue	Optional. Name of output queue that is used for MQ PUT operations.
MQ character set ID	Optional for client mode only. Coded Character Set Identification used by the RPC Server for IBM MQ.
MQ wait time	Optional. Wait interval for MQ GET operation in milliseconds.
MQ priority	Optional. Message priority for messages sent to MQ.
SSL cipher suite	Optional for client mode. Configuration for SSL connection to MQ server.
XML mapping	Optional. Name of XMM (XML Mapping) file if MQ message payload is XML/SOAP. If this is specified, messages to/from MQ will be converted to XML/SOAP.
MQ Java installation path	Required. MQ Java installation path. If set, take the value of the environment variable MQ_JAVA_INSTALL_PATH from the MQ installation.
MQ Java library path	Required for bindings mode only. MQ Java library path. If set, take the value of the environment variable MQ_JAVA_LIB_PATH from the MQ installation.
Credentials	
MQ user	Required if MQ server is running with security.
MQ password	Required if MQ server is running with security.
User Exit	
User exit	Optional. Class name for MQ server user exit.
User exit classpath	Optional. URL of the classpath for the user exit.

Server

Here you can specify the RPC Server settings.

Parameter	Description
RPC Server	
RPC Server address	Required. The case-sensitive RPC server address has the format: CLASS/SERVER/SERVICE.
Administration port	Required. The administration port in range from 1025 to 65535.
Reconnection attempts	Required. Number of reconnection attempts to the broker. When the number of attempts is reached and a connection to the broker is not possible, the RPC Server stops.
Worker Scalability	
Worker model	You can either have a fixed or dynamic number of workers. Default is dynamic (true). For more information see Worker Models .
Fixed number	Required. Fixed number of workers. Must be a number in range from 1 to 255.
Minimum number	Required. Minimum number of workers. Must be a number in range from 1 to 255.
Maximum number	Required. Maximum number of workers. Must be a number in range from 1 to 255.

Trace Level

Here you can set the trace level of the RPC Server for IBM MQ.

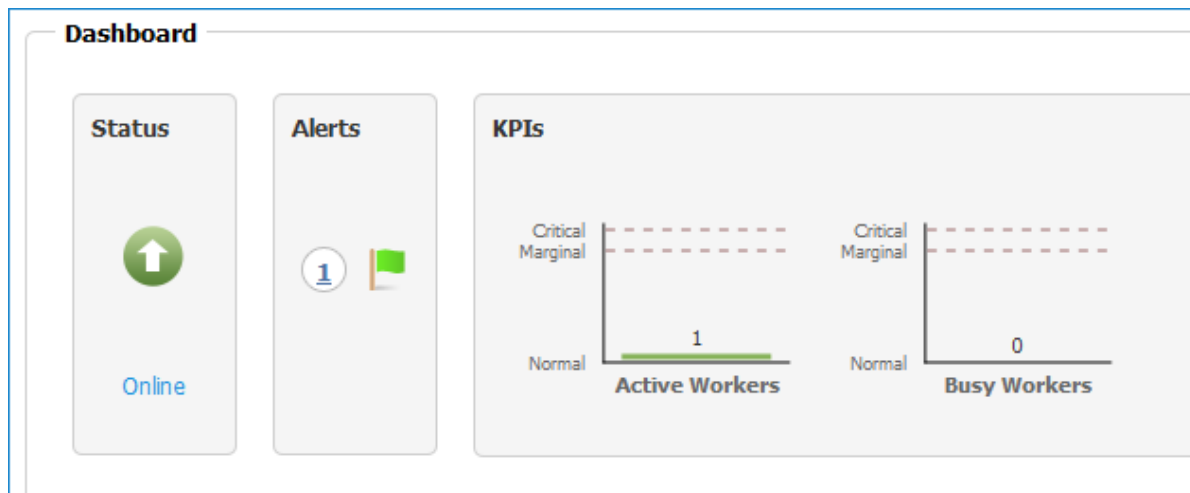
Parameter	Value	Description
Trace level	0 - 3	One of the following levels: 0 - None - No trace output (default). 1 - Standard - Minimal trace output. 2 - Advanced - Detailed trace output. 3 - Support - Support diagnostic. Use only when requested by Software AG support.

- 4 Click **Edit** to modify the parameters on your selected configuration type.
- 5 Click **Test** to check the correctness of your input or **Apply** to save your changes.

Viewing the Runtime Status

> To view the runtime status of the RPC server instance

- In the Command Central **Home** page, click the **Instances** tab and select the RPC Server for IBM MQ instance for which you want to see the runtime status (same as Step 1 under *Configuring a Broker Instance*).



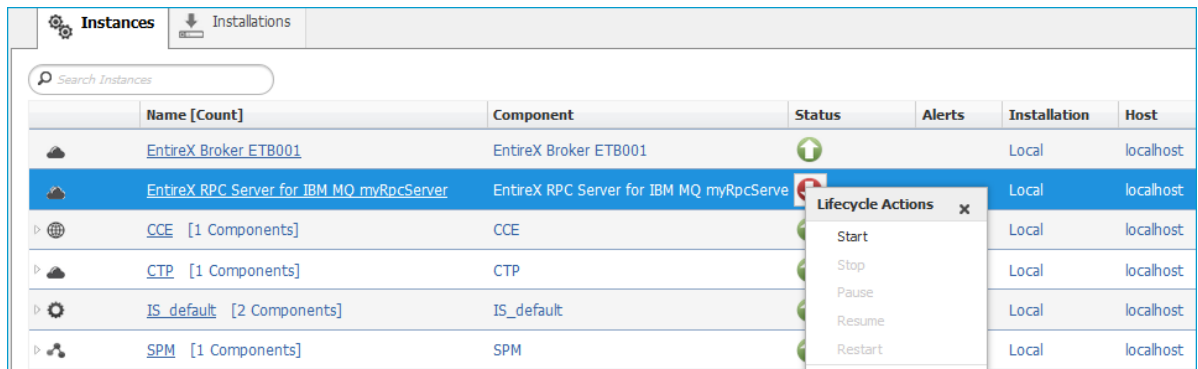
The visual key performance indicators (KPIs) and alerts enable you to monitor the RPC Server for IBM MQ's health.

KPI	Description
Active Workers	Number of active workers.
Busy Workers	Number of busy workers.

Starting an RPC Server Instance

➤ To start an RPC Server for IBM MQ instance from the Instances tab

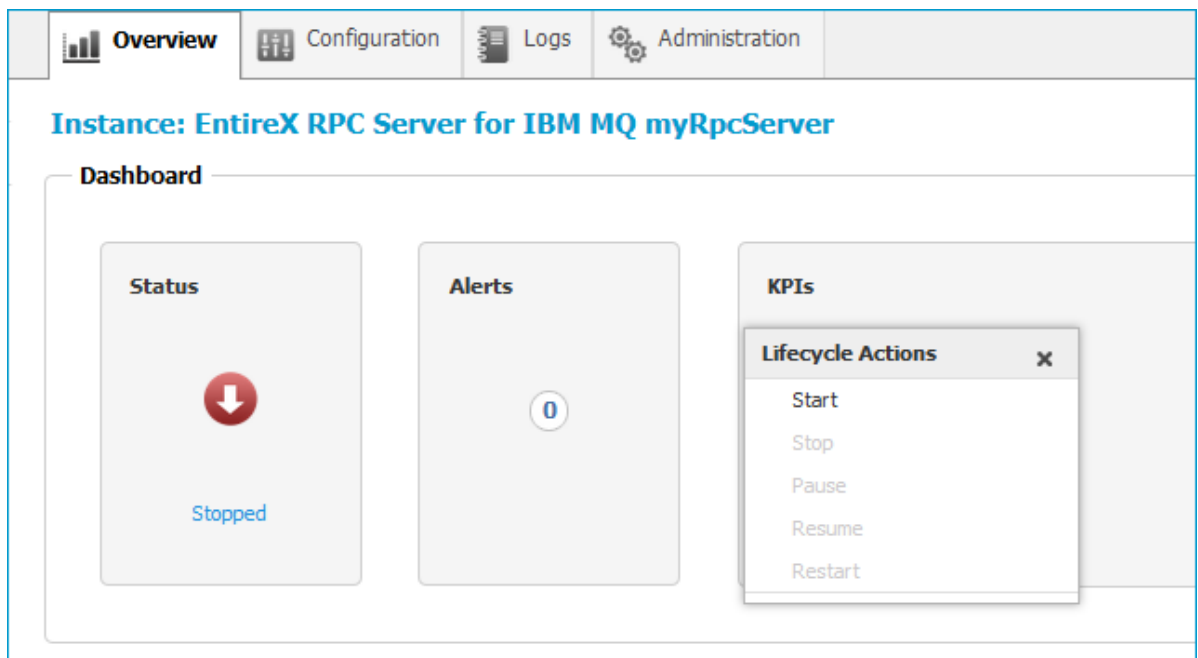
- 1 In the Command Central home page, click the **Instances** tab.



- 2 Select the status, and from the context menu choose **Start**.

➤ To start an RPC Server for IBM MQ instance from its Overview tab

- 1 In the Command Central home page, click the **Instances** tab and select the RPC Server for IBM MQ instance you want to start (same as Step 1 under *Configuring a Broker Instance*).

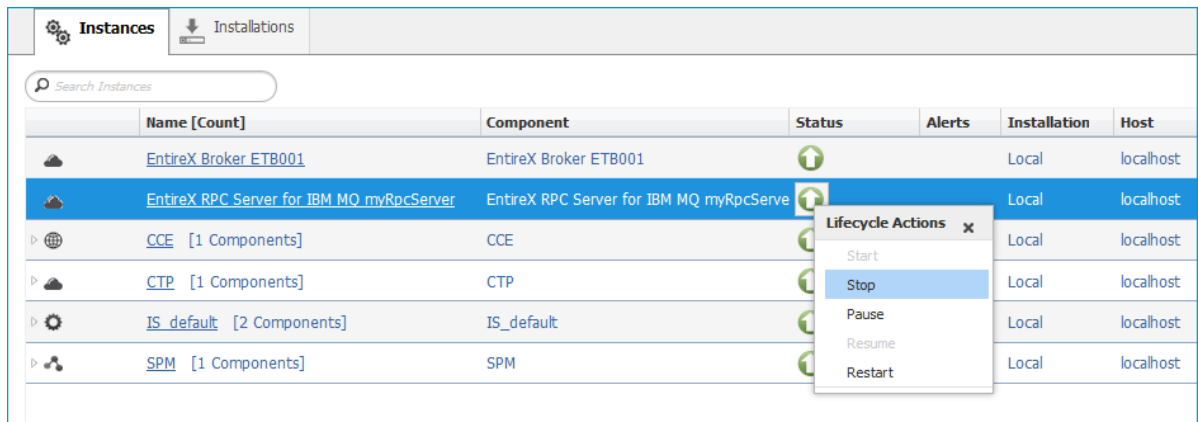


- 2 Select the status, and from the context menu choose **Start**.

Stopping an RPC Server Instance

➤ To stop an RPC Server for IBM MQ instance from the Instances tab

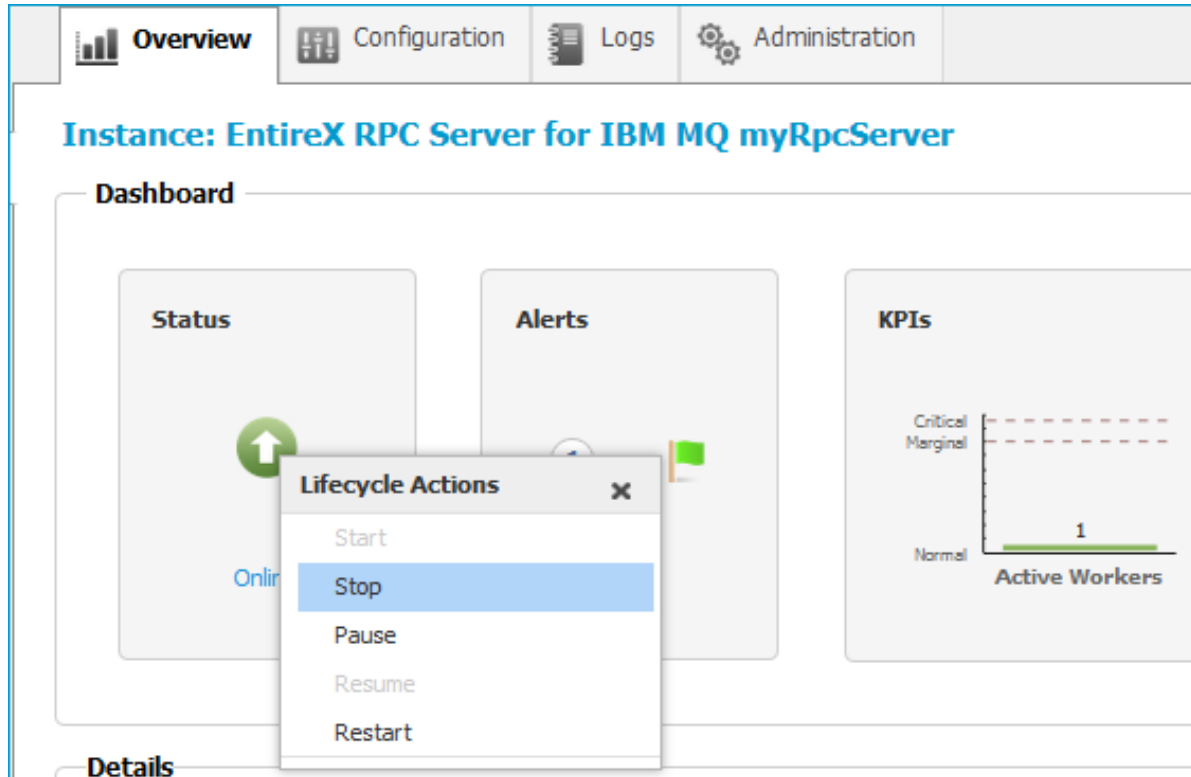
- 1 In the Command Central home page, click the **Instances** tab.



- 2 Select the status, and from the context menu choose **Stop**.

➤ To stop an RPC Server for IBM MQ instance from its Overview tab

- 1 In the Command Central home page, click the **Instances** tab and select the RPC Server for IBM MQ instance you want to stop (same as Step 1 under *Configuring a Broker Instance*).

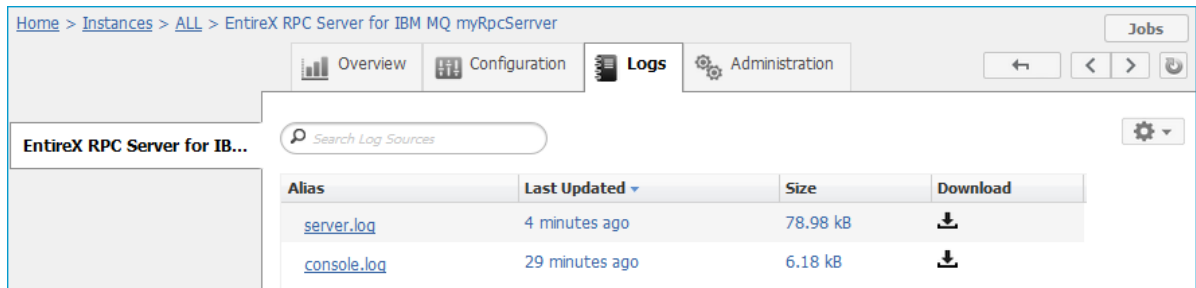


- 2 Select the status, and from the context menu choose **Stop**.

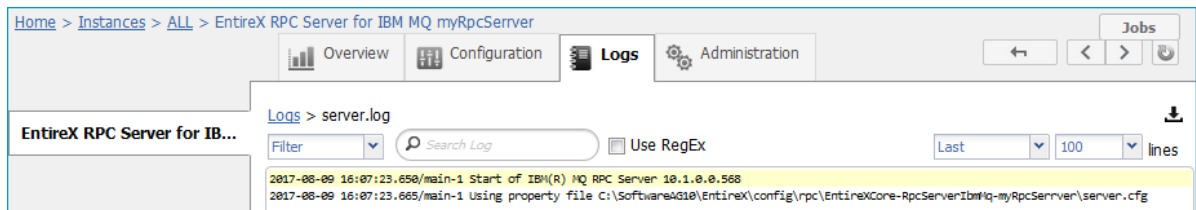
Inspecting the Log Files

➤ To inspect the log files of an RPC Server for IBM MQ instance

- 1 In the Command Central home page, click the **Instances** tab, then click the link associated with the RPC Server for IBM MQ instance for which you want to inspect the log files (same as Step 1 under *Configuring a Broker Instance*).
- 2 Click the **Logs** tab:



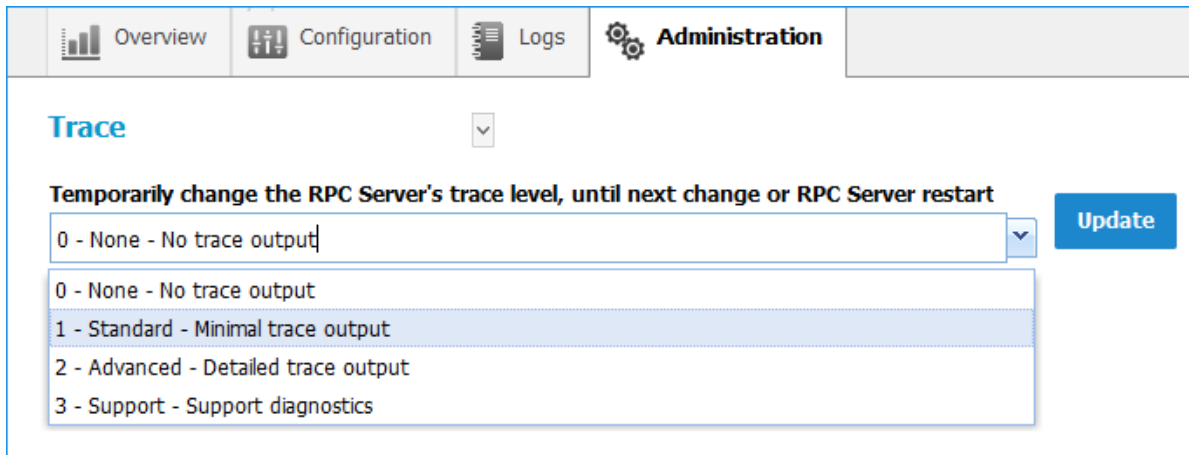
- 3 In the **Alias** column, click the link of the log file you want to inspect, for example *server.log*:



Changing the Trace Level Temporarily

➤ To temporarily change the trace level of an RPC Server for IBM MQ instance


- 1 In the Command Central home page, click the **Instances** tab then click the link associated with the RPC Server for IBM MQ instance for which you want change the trace level temporarily (same as Step 1 under *Configuring a Broker Instance*).
- 2 In the **Administration** tab, select the trace level and press **Update**.

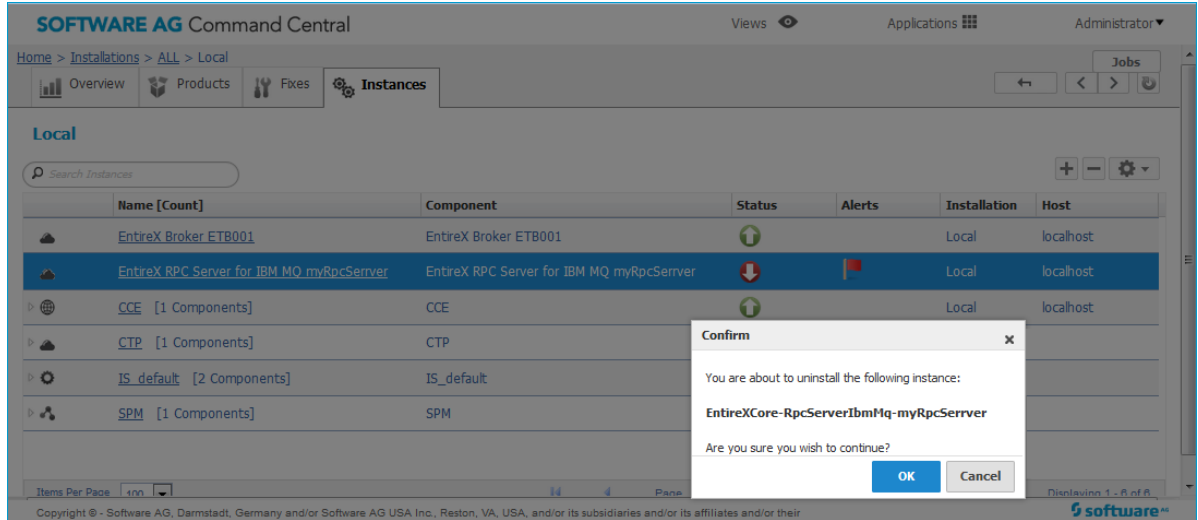


 **Note:** If you want to set the trace level permanently, see [Trace Level](#) under *Configuring an RPC Server Instance*.

Deleting an RPC Server Instance

➤ To delete an RPC Server for IBM MQ instance

- 1 In the list of EntireX RPC Server for IBM® MQ instances for your selected installation (for example Local), select the instance you want to delete and click the  button in the upper right corner above the list.



- 2 Click **OK** to confirm the uninstall of this RPC Server for IBM MQ instance.
- 3 In the next window, click **Finish**. The selected instance is removed from the list.

6 Administering the EntireX RPC Server for IBM® MQ

- Customizing the RPC Server 66
- Configuring the RPC Server Side 68
- Configuring the IBM MQ Side 71
- Starting the RPC Server 73
- Stopping the RPC Server 73
- Pinging the RPC Server 74
- Using SSL/TLS with the RPC Server 74
- Running an EntireX RPC Server as a Windows Service 75
- Activating Tracing for the RPC Server 76

The EntireX RPC Server for IBM® MQ runs as an RPC server and processes RPC client calls. It is used to send messages to and receive messages from an IBM MQ queue. This means that existing EntireX wrappers can be used for communication with IBM MQ.

This chapter describes how to administer the RPC Server for IBM MQ with local scripts as in earlier versions of EntireX.

See also *Administering the RPC Server for IBM MQ* with the Command Central [GUI](#) | [Command Line](#).

Customizing the RPC Server

The following elements are used to set up the RPC Server for IBM MQ:

- [Prerequisites](#)
- [Configuration File](#)
- [Start Script](#)

Prerequisites

There are two parts: the RPC server and the IBM® MQ side. The RPC Server for IBM MQ uses the following:

- The IBM® MQ base Java classes from IBM. To run the RPC Server for IBM MQ, you need either the base Java classes or a full installation of IBM® MQ. See *Prerequisites for RPC Server and Listener for IBM® MQ* in the respective section of the EntireX Release Notes for the required JAR file(s). The IBM® MQ environment variables `MQ_JAVA_LIB_PATH` and `MQ_JAVA_INSTALL_PATH` must be set. Make sure that either the local IBM® MQ installation or the IBM® MQ Java classes are accessible.
- The WS-Stack. Therefore the start scripts contain references to JAR files in the WS-Stack directory. If you update these JAR files, you may need to customize the JAR file names in the script files.

Configuration File

The default name for the configuration file is *entirex.wmqbridge.properties*. The RPC Server for IBM MQ 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`. Use the slash “/” as file separator. The configuration file contains the configuration for both parts of the RPC Server for IBM MQ.

Start Script

The start script for the RPC Server for IBM MQ is called *wmqbridge.bsh* (UNIX) or *wmqbridge.bat* (Windows) and is provided in the *bin* folder of the installation directory. You may customize this file. The script uses the configuration file *entirex.wmqBridge.properties* in the folder *etc*. The RPC Server for IBM MQ itself is contained in the file *entirex.jar*.

Configuring the RPC Server Side

The RPC server side of the RPC Server for IBM MQ is configured like the RPC Server for Java. The RPC Server for IBM MQ uses all properties starting 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.

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

With `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 you can use the command-line option. The command-line parameters have a higher priority than the properties set as Java system properties, and these have higher priority than the properties in the configuration file. For a list of all of the command-line parameters, use `-help`.

Property Name	Parameter	Default	Explanation																
<code>entirex.bridge.marshalling</code>			Define the type of marshalling (Natural or COBOL). Must be set only if the IDL file contains arrays of groups. See Mapping RPC Data to the MQ Message Buffer .																
<code>entirex.server.brokerid</code>	<code>-broker</code>	<code>localhost</code>	Broker ID. See <i>URL-style Broker ID</i> in the EntireX Broker ACI Programming documentation.																
<code>entirex.server.compresslevel</code>	<code>-compresslevel</code>	<code>0 (no compression)</code>	<table border="1"> <thead> <tr> <th colspan="2">Permitted values (you can enter the text or the numeric value):</th> </tr> </thead> <tbody> <tr> <td><code>BEST_COMPRESSION</code></td> <td><code>9</code></td> </tr> <tr> <td><code>BEST_SPEED</code></td> <td><code>1</code></td> </tr> <tr> <td><code>DEFAULT_COMPRESSION</code></td> <td><code>-1, mapped to 6</code></td> </tr> <tr> <td><code>DEFLATED</code></td> <td><code>8</code></td> </tr> <tr> <td><code>NO_COMPRESSION</code></td> <td><code>0</code></td> </tr> <tr> <td><code>N</code></td> <td><code>0</code></td> </tr> <tr> <td><code>Y</code></td> <td><code>8</code></td> </tr> </tbody> </table>	Permitted values (you can enter the text or the numeric value):		<code>BEST_COMPRESSION</code>	<code>9</code>	<code>BEST_SPEED</code>	<code>1</code>	<code>DEFAULT_COMPRESSION</code>	<code>-1, mapped to 6</code>	<code>DEFLATED</code>	<code>8</code>	<code>NO_COMPRESSION</code>	<code>0</code>	<code>N</code>	<code>0</code>	<code>Y</code>	<code>8</code>
Permitted values (you can enter the text or the numeric value):																			
<code>BEST_COMPRESSION</code>	<code>9</code>																		
<code>BEST_SPEED</code>	<code>1</code>																		
<code>DEFAULT_COMPRESSION</code>	<code>-1, mapped to 6</code>																		
<code>DEFLATED</code>	<code>8</code>																		
<code>NO_COMPRESSION</code>	<code>0</code>																		
<code>N</code>	<code>0</code>																		
<code>Y</code>	<code>8</code>																		

Property Name	Parameter	Default	Explanation
entirex.server.fixedservers		no	<p>NO The number of worker threads balances between what is specified in <code>entirex.server.minservers</code> and what is specified in <code>entirex.server.maxservers</code>. This is done by a so-called attach thread. At startup, the number of worker threads is the number specified in <code>entirex.server.minservers</code>. A new worker thread starts if the broker has more requests than there are worker threads waiting. If more than the number specified in <code>entirex.server.minservers</code> are waiting for requests, a worker thread stops if its receive call times out. The timeout period is configured with <code>entirex.server.waitserver</code>. See worker model DYNAMIC.</p> <p>YES The number of worker threads specified in <code>entirex.server.minservers</code> is started and the server can process this number of parallel requests. See worker model FIXED.</p>
	-help		Display usage of the command-line parameters.
entirex.server.logfile	-logfile		Name of the log file, the default is standard output.
entirex.server.maxservers		32	Maximum number of worker threads.
entirex.server.minservers		1	Minimum number of server threads.
entirex.server.password	-password		The password for secured access to the broker. The password is encrypted and written to the property <code>entirex.server.password.e</code> . To change the password, set the new password in the properties file. To disable password encryption, set <code>entirex.server.passwordencrypt=no</code> . Default: yes.
entirex.server.properties	-propertyfile	entirex.wmqbridge.properties	The file name of the property file.

Property Name	Parameter	Default	Explanation
entirex.server.restartcycles	-restartcycles	15	Number of restart attempts if the Broker is not immediately available. This can be used to keep the RPC Server for IBM MQ running while the Broker is temporarily down.
entirex.server.security	-security	no	no/yes/auto/Name of BrokerSecurity object.
entirex.server.serveraddress	-server	RPC/SRV1/CALLNAT	Server address.
entirex.server.serverlog	-serverlog		Name of the file where worker thread starts and stops are logged. Used by the Windows RPC Service.
entirex.server.userid	-user	WMQRPCServer	The user ID for access to the broker.
entirex.server.waitattach		600S	Wait timeout for the attach server thread.
entirex.server.waitserver		300S	Wait timeout for the worker threads.
entirex.timeout		20	TCP/IP transport timeout. See <i>Setting the Transport Timeout</i> under <i>Writing Advanced Applications - EntireX Java ACI</i> .
entirex.trace	-trace	0	Trace level (1,2,3).

Configuring the IBM MQ Side

These properties are used to configure the connection to the IBM® MQ queue manager.

Name	Parameter	Default Value	Explanation
entirex.wmqbridge. host			If host is not specified, bindings mode is used to connect to the local MQ Server. Otherwise specify the hostname or IP address of the MQ Server.
entirex.wmqbridge. port		1414	Port of the MQ Server. Not used in bindings mode.
entirex.wmqbridge. channel		SYSTEM.DEF. SVRCONN	Channel name used to the MQ Server. Not used in bindings mode.
entirex.wmqbridge. queuemanager	-wmqmanager		Name of the (local or remote) queue manager. If not specified, a connection is made to the default queue manager.
entirex.wmqbridge. inputqueue	-wmqinqueue		Name of input queue (the queue that is used for MQGET calls).
entirex.wmqbridge. outputqueue	-wmqoutqueue		Name of output queue (the queue that is used for MQPUT calls).
entirex.wmqbridge. userid	-wmquser		UserID for MQ Server.
entirex.wmqbridge. password	-wmqpassword		Password for MQ Server.
entirex.wmqbridge. waittime			Wait interval for MQGET call in milliseconds.
entirex.wmqbridge. userexit			Class name for user exit.
entirex.wmqbridge. userexit.classpath			URL of the classpath for user exit (optional).
entirex.wmqbridge. ccsid		platform encoding	Coded Character Set Identification used by the RPC Server for IBM MQ (which acts as an MQ client), unused in bindings mode. Enable character conversion in the broker by setting the service-specific attribute CONVERSION to "SAGTRPC". See also <i>Configuring ICU Conversion</i>

Name	Parameter	Default Value	Explanation
			under <i>Configuring Broker for Internationalization</i> in the platform-specific Administration documentation. More information can be found under <i>Internationalization with EntireX</i> .
entirex.wmqbridge.mqtrace		0	MQ tracing enabled if parameter > 0.
entirex.bridge.xmm			Name of XMM (XML Mapping) file; if MQ message payload is XML/SOAP. If this is specified, messages to/from MQ will be converted to XML.
entirex.bridge.xml.encoding		utf-8	Encoding of the XML document that is sent to MQ (if <code>entirex.bridge.xmm</code> is used).
entirex.wmqbridge.environment.sslCipherSuite			Configuration for SSL connection to MQ Server. See the IBM® MQ documentation for details.
entirex.wmqbridge.environment.sslFipsRequired			Configuration for SSL connection to MQ Server. See the IBM® MQ documentation for details.
entirex.wmqbridge.priority			Message priority for messages sent to MQ (different from the default priority of the destination queue).

The RPC Server for IBM MQ can be run to

- only send messages to MQ (only output queue specified),
- only receive messages from MQ (only input queue specified), or
- transport messages in both directions (bidirectional communication).

If your programs use arrays of groups, you have to set `entirex.bridge.marshalling` to "Natural" or "COBOL". If your programs do not use arrays of groups, you must not set `entirex.bridge.marshalling`.

Alternatively the RPC data can be transformed to/from XML or SOAP as defined by an XMM mapping file from the XML/SOAP Wrapper. To achieve this, specify the parameter `entirex.bridge.xmm`.

Starting the RPC Server

➤ To start the RPC Server for IBM MQ

- Use the *Start Script*.

Or:

Under Windows you can use the RPC Server for IBM MQ as a Windows Service. See *Running an EntireX RPC Server as a Windows Service*.

Stopping the RPC Server

➤ To stop the RPC Server for IBM MQ

- 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`.

Pinging the RPC Server

> To ping the RPC Server for IBM MQ

- 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.

Using SSL/TLS with the RPC Server

To use SSL with RPC Server for IBM MQ, you need to configure two sides:

■ IBM® MQ Side

See parameters `entirex.wmqbridge.environment.sslCipherSuite` and `entirex.wmqbridge.environment.sslFipsRequired` under [Configuring the IBM MQ Side](#).

■ RPC Server side

On the RPC server side you 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. On the RPC server side, an SSL client is configured. 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 Set up the RPC server side for IBM MQ for an SSL connection.

Use the *URL-style Broker ID* with protocol `ssl://` for the Broker ID. If no port number is specified, port 1958 is used as default. Example:

```
ssl://localhost:22101?trust_store=C:\SoftwareAG\EntireX\etc\ExxCACert.jks&verify_server=no
```

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*.

- 3 Make sure the SSL server to which the RPC side 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)

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 IBM MQ as a Windows Service

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



Note: The script must pass external parameters to the RPC server and use the reduced signaling of the JVM (option `-Xrs`):

```
java -Xrs com.softwareag.entirex.rpcbridge.WMQBridge %*
```

If `-Xrs` is not used, the JVM stops and an entry 10164002 is written to the event log when the user logs off from Windows.

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 `wmqbridge.bat`, the command will be

```
RPCService -install -ext MyServer -script install_path\EntireX\bin\wmqbridge.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

The trace level for the RPC server side is controlled by the usual `entirex.trace` property.

Tracing of the IBM MQ classes can be influenced with the property `entirex.wmqbridge.mqtrace`.

Redirect the trace to a file with the property `entirex.server.logfile`. Set this to the file name of the log file, default is standard output.

All properties can be set in the configuration file.

7 Advanced RPC Server for IBM MQ Functionality

▪ Support for Request/Reply Scenarios	78
▪ Handling of Correlation ID	78
▪ Character Encoding Issues	78
▪ User Exit for Message Processing	79
▪ Transactional Behavior	80

Support for Request/Reply Scenarios

A synchronous request/reply call to MQ is possible. In this case, the remote procedure call has to have both `INPUT` and `OUTPUT` parameters, or an `INOUT` parameter has to be specified. The RPC Server for IBM MQ issues an `MQ PUT` call with message type "Request" on the default output queue. The `Reply To Queue Name` field is set to the name of the default input queue. After the `MQPUT` call, the RPC Server for IBM MQ issues an `MQGET` call on the default input queue and then it waits for the reply message (note that for this scenario the input queue must be different from the output queue).

The request and reply messages are correlated by the correlation ID. The reply message has to have the same correlation ID as the request message.

Handling of Correlation ID

For Request Reply scenarios there is an implicit usage of the correlation ID by the RPC Server for IBM MQ: MQ is instructed to generate a correlation ID. The option `MQPMO_NEW_CORREL_ID` is set internally to achieve this. When reading the reply the option `MQMO_MATCH_CORREL_ID` is specified, thus the reply message has to use the same correlation as specified in the request message.

Character Encoding Issues

When the RPC Server for IBM MQ is exchanging messages via the EntireX Broker with an RPC client, the usual rules apply. By default, the message is exchanged between the RPC Server for IBM MQ and EntireX using the platform encoding of the JVM that executes the RPC Server for IBM MQ.

If the payload of the MQ message is in XML format (property `entirex.bridge.xmm` has been set), the RPC Server for IBM MQ converts the XML payload to the encoding used for the remote procedure call. If the RPC Server for IBM MQ has to create the XML payload, it will use UTF-8. A different encoding can be used by setting the property `entirex.bridge.xml.encoding`.

If the payload of the MQ message is of type text, the translation of the MQ message payload is done by the IBM MQ Java classes. When sending a message, the RPC Server for IBM MQ converts the message to the encoding specified by the CCSID (Coded Character Set IDentification) of the queue manager. When receiving a message, the RPC Server for IBM MQ converts the message to the platform encoding of the JVM.



Note: The default platform encoding of the JVM can be changed by setting the system property `file.encoding` in the startup script of the RPC Server for IBM MQ.

User Exit for Message Processing

IBM® MQ does not have a clearly defined message layout, it is basically a stream of bytes. In general it is up to the MQ application to know the exact semantics of an MQ message. This might include application-specific headers and formatting rules. The RPC Server for IBM MQ supports a general but simplified model of message processing.

To better handle application specific message layout details a user exit (or callback routine) can be used. The user exit is working on the IBM® MQ Java representation of an MQ message (class `com.ibm.mq.MQMessage`) and can change the MQ message. The user exit gets control:

1. after an MQ message has been constructed by the RPC Server for IBM MQ and before the message is put to the MQ queue,
2. after an MQ message has been read from an MQ queue and before it is processed by the RPC Server for IBM MQ.

The user exit can be used, for example, for an application specific processing of the `MQRFH`, `MQRFH2` or even custom headers.

To enable a user exit, use the property `entirex.wmqbridge.userexit` to specify the class name of the user exit implementation. The class will be loaded using the standard classpath. You can specify a separate classpath with the property `entirex.wmqbridge.userexit.classpath`. Note that for the classpath a file or HTTP URL must be specified. Your user exit class must implement the Java interface `com.softwareag.entirex.rpcbridge.WMQBridgeExit`. This Java interface has the following methods:

```
/**
 * This method is called after the message has been created by the WMQBridge
 * and before the message is sent to an MQ queue (MQPUT).
 * The Message object and/or the MessageOptions object can be changed.
 */
/**
 * @param msg The MQ message object.
 * @param pmo The MQPutMessageOptions object.
 */
public void beforePut(com.ibm.mq.MQMessage msg, com.ibm.mq.MQPutMessageOptions pmo);
/**
 * This method is called before a message is retrieved from an
 * MQ queue (MQGET). The MessageOptions object can be changed.
 */
/**
 * @param gmo The MQGetMessageOptions object.
 */
public void beforeGet(com.ibm.mq.MQGetMessageOptions gmo);
/**
 * This method is called after a message has been retrieved from an
 * MQ queue (MQGET) and before the message will be processed by the WMQBridge.
 * The Message object can be changed.
 */
```

```
**  
** @param msg The MQ message object.  
**/  
public void afterGet(com.ibm.mq.MQMessage msg);
```

Transactional Behavior

Calls to MQ Series are non-transactional by default. Thus the request operates outside the normal unit-of-work protocols. When reading a message with `MQGET`, the message is deleted from the queue immediately. If an error occurs in the further processing of the message within the RPC Server for IBM MQ (for example the translation to RPC or XML results in an error), the message cannot be made available again. The same applies to sending a message, the `MQPUT` call makes the message available immediately.

If the RPC client application uses conversational RPC, the MQ calls are issued transactional (using the `SYNCPOINT` option). A Backout Conversation will send a backout to the queue manager, and a Commit Conversation will send a commit to the queue manager.

To understand the level of guaranteed delivery provided by the RPC Server for IBM MQ we present the flow of control when reading a message from a queue or writing a message to a queue. Sending a message to an MQ queue:

➤ To send a message to an MQ Queue

- 1 The RPC client application sends a send request to the RPC Server for IBM MQ.
- 2 The RPC Server for IBM MQ creates a corresponding MQ message and puts the message on the queue. If the remote procedure call is part of an RPC conversation, the message is not committed.
- 3 The RPC Server for IBM MQ returns a positive acknowledgment back to the RPC client. If something fails in step 2, an error is returned to the RPC client.

➤ To receive a message from an MQ Queue

- 1 The RPC client application sends a receive request to the RPC Server for IBM MQ.
- 2 The RPC Server for IBM MQ reads a message from the queue. If no message is available, an error is returned to the RPC client. If the remote procedure call is part of an RPC conversation, the message is not committed.
- 3 The RPC Server for IBM MQ creates a corresponding RPC reply that is sent back to the RPC client. If something fails in step 2, an error is returned to the RPC client.

If the send or receive call is part of a conversational RPC, the MQ transaction will get a commit or backout when the RPC conversation is closed, depending on the type of the `endConversation` call.