webMethods EntireX

WMQ RPC Server and Listener

Innovation Release

Version 9.9

October 2015

WEBMETHODS

This document applies to webMethods EntireX Version 9.9 and all subsequent releases.

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

Document ID: EXX-WMQ-99-20171128BRIDGE

Table of Contents

1 Introduction to the WebSphere MQ RPC Server and WebSphere MQ Listener

Overview

The EntireX WebSphere MQ RPC Server allows standard RPC clients to send and receive asynchronous and synchronous messages to a WebSphere MQ queue manager. The EntireX WebSphere MQ Listener receives asynchronous and synchronous messages from a WebSphere MQ queue and calls a standard RPC server.

Both components use the WebSphere MQ base Java classes from IBM.

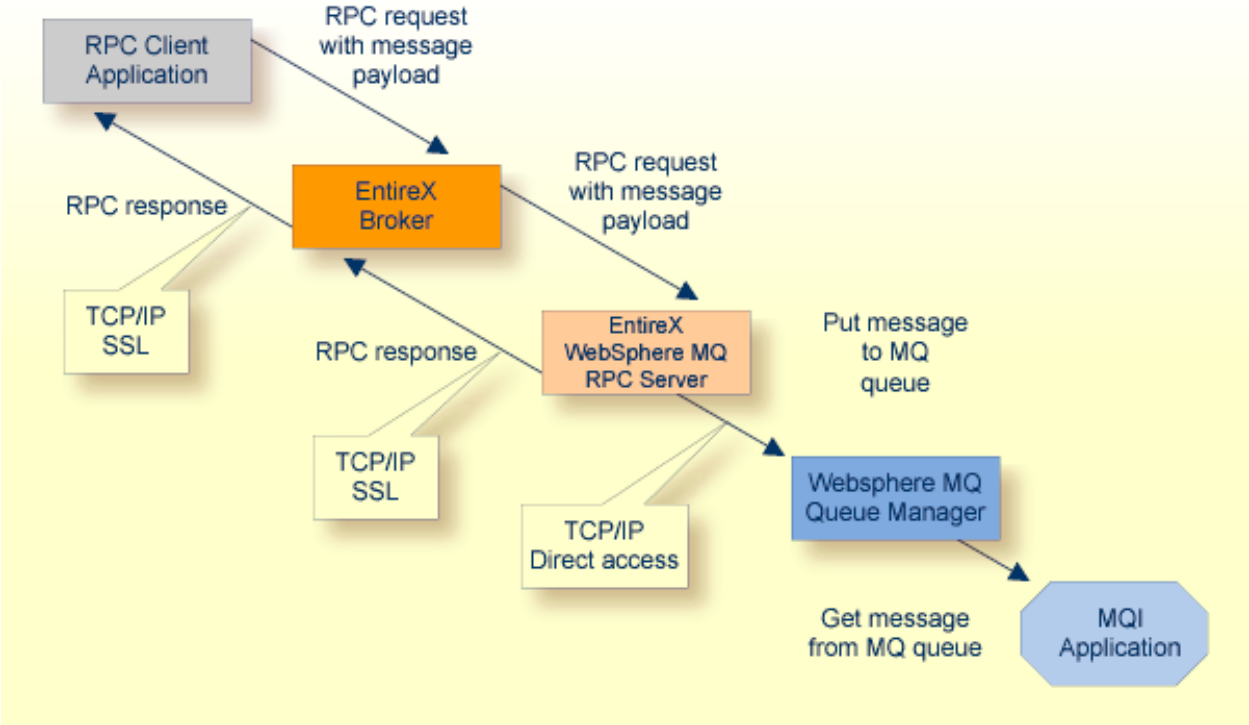
Both components can connect to a WebSphere MQ either as a WebSphere MQ client using TCP/IP (*client mode*) or in so-called *bindings mode* where it is connected directly to WebSphere MQ running on the same machine. Note that on z/OS, only bindings mode is supported. If the WebSphere MQ Listener wants to connect in client mode via TCP/IP to an MQ server on z/OS, the *client attachment feature* needs to be installed on the target queue manager.

The WebSphere MQ RPC Server runs as an RPC server and processes RPC client calls. An RPC client can send an asynchronous message (MQ `PUT` 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 (MQ `GET` 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. 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 WebSphere MQ RPC Server registers to one RPC service. On the MQ side it uses one input queue and one output queue.

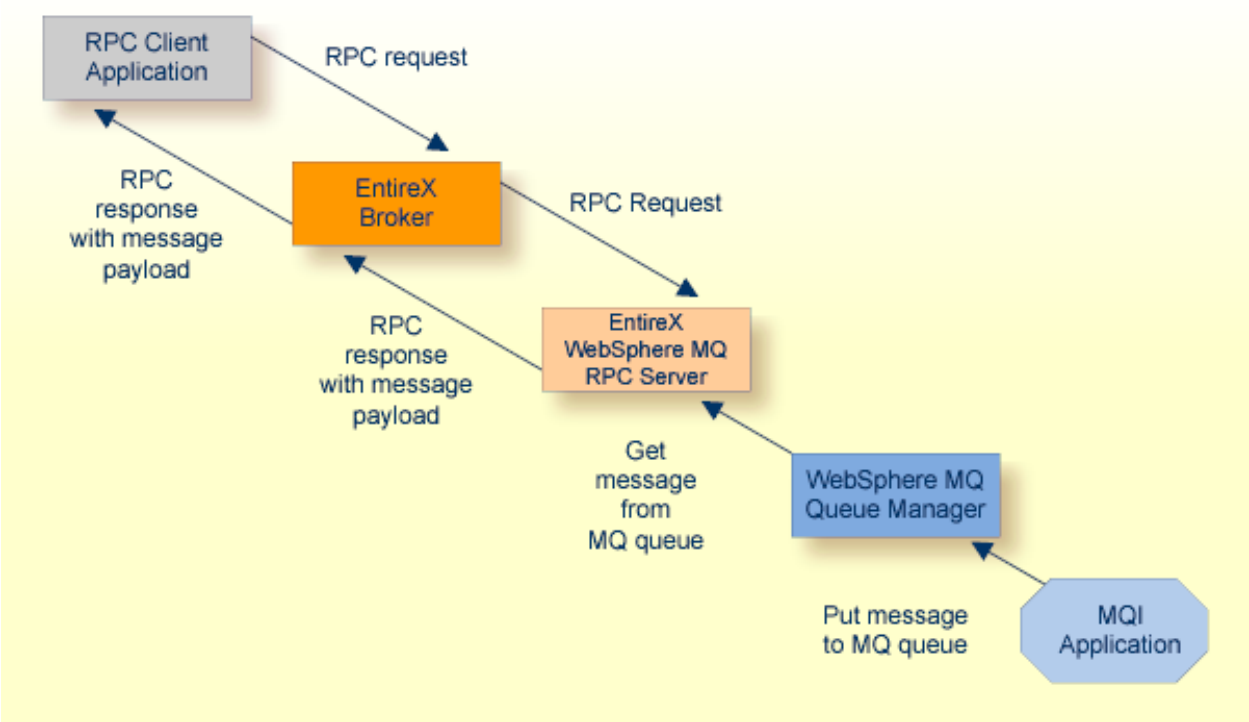
The WebSphere MQ Listener runs as a listener on an MQ queue and processes MQ messages. It receives an MQ message and sends the message to an RPC server. A synchronous scenario is possible if the MQ message is a request message that specifies a reply queue. In this case the result returned by the RPC server is sent back as an MQ message to the reply queue.

Sending a Message to a WebSphere MQ Queue



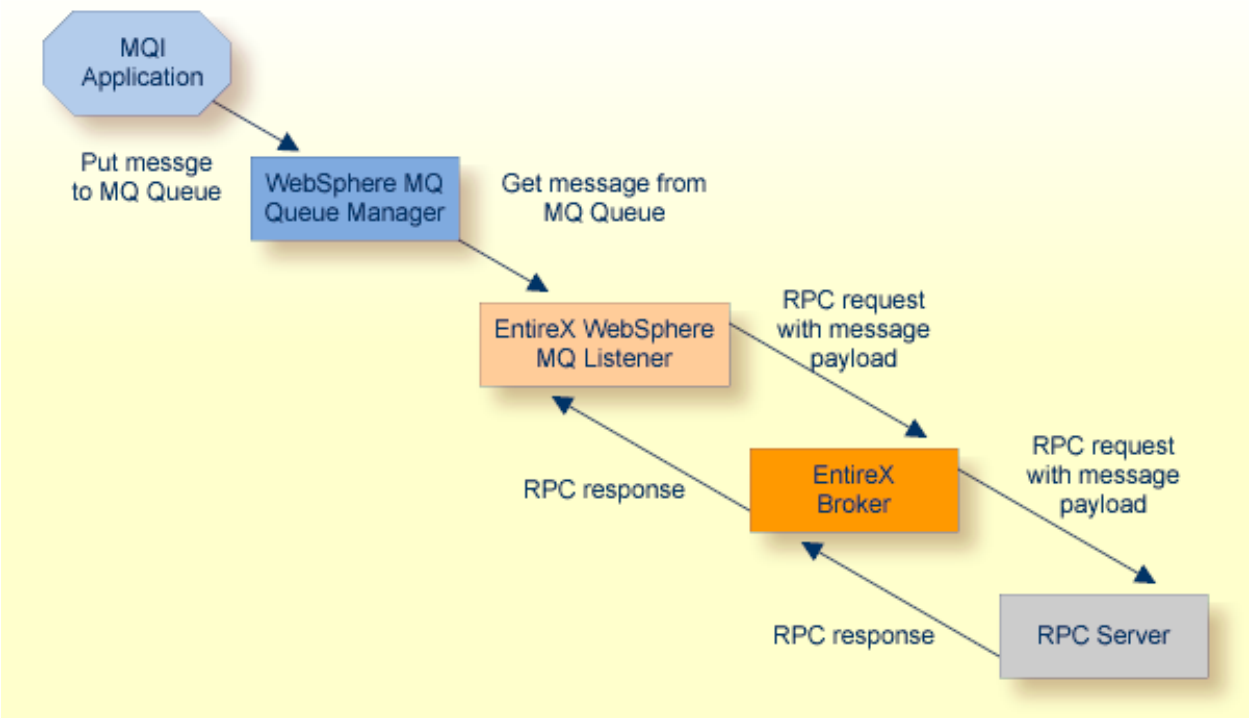
Note: All messages sent to a WebSphere MQ RPC Server instance via a specific RPC service are put on the same MQ output queue.

Receiving a Message from a WebSphere MQ Queue



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

Receiving a Message from a WebSphere MQ Queue via WebSphere MQ Listener



Note: All messages retrieved by the WebSphere MQ Listener from the MQ listen queue are passed to the same RPC service. Messages are retrieved in the order they appear on the queue.

2 Administering the EntireX WebSphere MQ RPC Server

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

Customizing the WebSphere MQ RPC Server

To set up the WebSphere MQ RPC Server, there is a configuration file and there are scripts to start the WebSphere MQ RPC Server.

The WebSphere MQ RPC Server is contained in *entirex.jar*. There are two parts: the RPC server and the WebSphere MQ side.

The WebSphere MQ RPC Server uses the WebSphere MQ base Java classes from IBM. To run the WebSphere MQ RPC Server, you need either the base Java classes or a full installation of WebSphere. Prerequisites for all EntireX components are described centrally. See *Prerequisites for WebSphere MQ RPC Server and WebSphere MQ Listener* in the respective section of the Release Notes for the required JAR file(s). The WebSphere MQ environment variables `MQ_JAVA_LIB_PATH` and `MQ_JAVA_INSTALL_PATH` must be set.

Make sure that either the local WebSphere MQ installation or the WebSphere MQ Java classes are accessible.

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

Use the RPC server agent of the System Management Hub for setup. Add the WebSphere MQ 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.

Alternatively, you can use a script to start the WebSphere MQ RPC Server. On Windows, use *wmqbridge.bat* in the folder *bin* to start the WebSphere MQ RPC Server. On UNIX, use *wmqbridge.bsh* in the folder *bin* to start the WebSphere MQ RPC Server. Both scripts use the configuration file *entirex.wmqBridge.properties* in the folder *etc*, and both can be customized.

Configuring the RPC Server Side

The RPC server side of the WebSphere MQ RPC Server is configured like the Java RPC Server. The WebSphere MQ RPC Server 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`.

Name	Parameter	Default Value	Explanation
`entirex.bridge.marshalling`			Define the type of marshalling (Natural or COBOL). Must be set only if the IDL file contains arrays of groups. See ***Mapping IDL Data Types to the MQ Message Buffer***.
`entirex.server.brokerid`	`-broker`	`localhost`	Broker ID. See *URL-style Broker ID*.
`entirex.server.compresslevel`	`-compresslevel`	`0` (no compression)	Permitted values (you can enter the text or the numeric value):
			`BEST_COMPRESSION` → `9`
			`BEST_SPEED` → `1`
			`DEFAULT_COMPRESSION` → `-1`, mapped to `6`
			`DEFLATED` → `8`
			`NO_COMPRESSION` → `0`
			`N` → `0`
			`Y` → `8`

Name	Parameter	Default Value	Explanation
entirex.server.encryptionlevel	-encryption	0	Deprecated. For encrypted transport we strongly recommend using the Secure Sockets Layer/Transport Layer Security protocol. See *SSL/TLS and Certificates with EntireX*.
entirex.server.fixedservers		no	If no, use an attach server thread to manage worker threads, otherwise run the minimum number of server threads. See the properties entirex.server.maxservers and entirex.server.minservers.
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.monitorport	-smhport	0	The port where the server listens for commands from the System Management Hub (SMH). If this port is 0, no port is used and management by the SMH is disabled.
entirex.server.password	-password		The password for secured access to the Broker. The password is encrypted and written to the property entirex.server.password.e. To change the password, set the new password in the properties file (default is *entirex.wmqbridge.properties*). To disable password encryption, set entirex.server.passwordencrypt=no (default for this property is yes).
entirex.server.properties	-propertyfile	entirex.wmqbridge.properties	The file name of the property file.
entirex.server.restartcycles	-restartcycles	15	Number of restart attempts if the Broker is not immediately available. This can be used to keep the Java RPC Server 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.

Name	Parameter	Default Value	Explanation
entirex.server.userid	-user	WMQRPCServer	The user ID for the Broker for RPC. See entirex.server.password.
entirex.server.verbose	-verbose	no	Enable verbose output to the log file.
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 *Setting the Transport Timeout* under *Writing Advanced Applications - EntireX Java ACI*.
entirex.trace	-trace	0	Trace level (1,2,3).

Configuring the WebSphere MQ Side

These properties are used to configure the connection to the WebSphere 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	-wmqqmanager		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 which is used for MQ GET operations).
entirex.wmqbridge. outputqueue	-wmqoutqueue		Name of output queue (the queue which is used for MQ PUT operations).
entirex.wmqbridge. userid	-wmquser		UserID for MQ Server.
entirex.wmqbridge. password	-wmqpassword		Password for MQ Server.
entirex.wmqbridge. waittime			Wait interval for MQ Get operation in milliseconds.
entirex.wmqbridge. userexit			Class name for WMQBridge user exit.
entirex.wmqbridge. userexit. classpath			URL of the classpath for WMQBridge user exit (optional).
entirex.wmqbridge. ccsid		platform encoding	Coded Character Set Identification used by the EntireX WebSphere MQ RPC Server (which acts as an MQ client), unused in bindings mode.
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.

Name	Parameter	Default Value	Explanation
`entirex.bridge.xml. encoding`		`utf-8`	Encoding of the XML document which is sent to MQ (if entirex.bridge.xmm is used).
`entirex.bridge. names.file`			Name of a properties file generated with the bridge.tpl template which contains names of the first level parameters in the IDL file (optional). Necessary if IDL file contains dynamic MQ parameters.
`entirex.bridge. verbose`		`no`	Verbose/trace mode of WebSphere MQ RPC Server
`entirex.wmqbridge. environment. sslCipherSuite`			Configuration for SSL connection to MQ Server. See the WebSphere MQ documentation for details.
`entirex.wmqbridge. environment. sslFipsRequired`			Configuration for SSL connection to MQ Server. See the WebSphere MQ documentation for details.
`entirex.wmqbridge.priority`			Message priority for messages sent to MQ (different from the default priority of the destination queue).

The WebSphere MQ RPC Server 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`.

Mapping IDL Data Types to the MQ Message Buffer

The WebSphere MQ RPC Server uses a predefined mapping of IDL data types to the MQ message buffer.

Data Type	Description	Format	Note
A*number*	Alphanumeric	*number* bytes, encoding the characters.	
AV	Alphanumeric variable length	Bytes up to the end of the buffer.	1, 4
AV[*number*]	Alphanumeric variable length with maximum length	Bytes up to the end of the buffer, maximal length *number*.	1
K*number*	Kanji	Same as data type A.	
KV	Kanji variable length	Same as data type AV.	1, 4
KV[*number*]	Kanji variable length with maximum length	Same as data type AV[*number*].	1
I1	Integer (small)	*sign* (+, -) and 3 bytes (digits).	
I2	Integer (medium)	*sign* (+, -) and 5 bytes (digits).	
I4	Integer (large)	*sign* (+, -) and 10 bytes (digits).	
N*number1*[.*number2*]	Unpacked decimal	*sign* (+, -), *number1* bytes (digits) [*number2*] bytes (digits), no decimal point.	
P*number1*[.*number2*]	Packed decimal	*sign* (+, -), *number1* bytes (digits) [*number2*] bytes (digits), no decimal point.	
L	Logical	1 byte: X for true, all other false.	
D	Date	*YYYYMMDD*.	2
T	Time	*YYYYMMDDhhmmssS*.	3

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.

Data types not supported:

- Binary (B[*n*],BV, BV[*n*])
- Floating point (F4, F8)

Starting the WebSphere MQ RPC Server

Use start script `wmqbridge.bsh` (UNIX) or `wmqbridge.bat` (Windows) in the folder *bin* to start the WebSphere MQ RPC Server. You may customize this file. See also *Prerequisites for WebSphere MQ RPC Server and WebSphere MQ Listener* in the respective section of the Release Notes.

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.

You can use the RPC server agent in the System Management Hub to configure and start the WebSphere MQ RPC Server.

Under Windows you can start the WebSphere MQ RPC Server as a Windows Service. The installation of the service is similar to the installation of the Java RPC Server as Windows Service. See *Running the Java RPC Server as a Windows Service* under under *Administering the EntireX Java RPC Server* in the Windows administration documentation in the Windows administration documentation.

Stopping the WebSphere MQ RPC Server

Use the RPC server agent in the System Management Hub to stop the WebSphere MQ RPC Server. You can also stop the WebSphere MQ RPC Server with CTRL-C.

On UNIX you can use `kill <pid of Java process>` to stop the WebSphere MQ RPC Server.

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

Using SSL/TLS

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

- **WebSphere MQ Side**
  See parameters `entirex.wmqbridge.environment.sslCipherSuite` and `entirex.wmqbridge.environment.sslFipsRequired` under ***Configuring the WebSphere MQ Side***.

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

2 Set up the WebSphere MQ RPC Server 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 EntireX Clients and Servers*.

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

- *Settting up and Administering the EntireX Broker SSL Agent* in the UNIX and Windows administration documentation
- *Support for SSL/TLS* in the EntireX Adapter documentation (for Direct RPC)

Tracing the WebSphere MQ RPC Server

The trace level for the EntireX RPC Server part is controlled by the usual `entirex.trace` property. It can be set in the configuration file. Additional diagnostic output can be enabled by setting the property `entirex.server.verbose`.

The WebSphere MQ RPC Server specific diagnostic output is enabled by setting the property `entirex.bridge.verbose`. In addition, tracing of the IBM WebSphere MQ classes can be influenced with the property `entirex.wmqbridge.mqtrace`.

Use the RPC server agent of the System Management Hub to dynamically change the level of the diagnostic output. You can specify a value of 0, 1, 2, or 3 which changes the value of `entirex.trace`. In addition, the value 0 will disable both `entirex.server.verbose` and `entirex.bridge.verbose`. A value greater than 0 will enable both `entirex.server.verbose` and `entirex.bridge.verbose`.

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.

3 Advanced WebSphere MQ RPC Server Functionality

Support for Dynamic Queue Names

In the properties file, only one input and one output queue can be specified. These are the default queues used by the WebSphere MQ RPC Server. The queue name can also be specified on each call, using a dynamic parameter. Since different queue names can be specified on different calls, multiple input or output queues are supported. The default input and output queues can also be used with the dynamic parameter. Queues specified by a dynamic parameter are opened when they are used for the first time and closed when the WebSphere MQ RPC Server terminates.

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 WebSphere MQ RPC Server 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 `PUT` call, the WebSphere MQ RPC Server issues an MQ `GET` 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).

Dynamic queue names can also be used for the request/reply scenario. Use the `MQ_QUEUE_NAME` parameter for the output (request) queue and the `MQ_REPLY_QUEUE` parameter for the input (reply) queue.

You can also use the dynamic `Reply To Queue` parameter to indicate that a reply is expected for this message. In this case, sending the message and receiving the reply is decoupled and is performed by two separate RPC requests.

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.

Dynamic IDL/RPC Parameters for WebSphere MQ RPC Server

With the WebSphere MQ RPC Server it is possible that certain parameters of a remote procedure call are dynamic parameters which are evaluated by the WebSphere MQ RPC Server. Dynamic parameters have a fixed name; they can be defined only on level 1 in the parameter definition in the IDL file and before any variable length parameter, and have a specific format. The WebSphere MQ RPC Server uses the following parameters:

Parameter	Format	Description
Queue name (MQ_QUEUE_NAME)	A48	Overrides the default input or output queue name specified in the properties file. IN parameter only.
Correlation ID (MQ_CORREL_ID)	A48	The MQ Correlation ID can be used in Request Reply scenarios. If it is an IN (or INOUT) parameter, it is assigned to the correlationId parameter of the MQ message which is sent to MQ. If it is an OUT (or INOUT) parameter, it contains the corresponding value of the received MQ message. This parameter is defined of type alphanumeric but its contents is an hexadecimal encoded binary array.
Reply To Queue (MQ_REPLY_QUEUE)	A48	The replyToQueueName parameter of the MQ message.If it is an IN (or INOUT) parameter, it specifies that the MQ message is of type Request. If it is an OUT (or INOUT) parameter, it contains the corresponding value of the received MQ message.
Wait Interval (MQ_WAIT_INTERVAL)	A with fixed length	The wait interval in milliseconds for an MQ GET call. Overrides the default value entirex.wmqbridge.waittime from the properties file. IN parameter only.
MQRFH Header data (MQ_RFH_*)	A with fixed length	Arbitrary number of parameters. The names following the prefix "MQ_RFH_" are used as the names of the name value pairs of the MQRFH header.

If dynamic parameters are to be used, generate a properties file from the corresponding Software AG IDL file and specified with the entirex.bridge.names.file property. To generate this property file, use the template *bridge.tpl* in the *template* subdirectory of the EntireX installation. For batch generation, run erxidl.bat (Windows) or erxidl.bsh (UNIX) with the parameters "-t <*path to template directory*>/bridge.tpl <*idlFile*>".

Alternatively, you can also use the *EntireX Workbench*. Go to the **Preferences for EntireX** and create a new custom wrapper. Specify a name and browse to the *bridge.tpl* template. If the Custom Wrapper has been created (and the Workbench restarted), you can generate the properties file from an IDL file, using the context menu item **Other > Generate...**.

Handling of Correlation ID

The correlation ID can be explicitly used in get message and put message operations using the dynamic parameter MQ_CORREL_ID.

For Request Reply scenarios there is also an implicit usage of the correlation ID by the WebSphere MQ RPC Server: If MQ_CORREL_ID is not explicitly specified, 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.

Support for the MQRFH Header

MQ messages may have custom specific headers. The `MQRFH` header (rules and formatting header) consists basically of name value pairs. Restriction: only one header per MQ message is possible; MQ allows an arbitrary number of headers per message.

When sending a message to MQ: a `MQRFH` header is built if at least one parameter in the IDL file has a name with prefix `"MQ_RFH_"`. All IN (or INOUT) parameters with this prefix are used to build the header. If for example the IDL file contains two fields `MQ_RFH_H1` and `MQ_RFH_H2` with the values `v1` and `v2`, the resulting MQRFH header will have two name value pairs, `H1 v1` and `H2 v2`.

If a message is received from MQ: if the message has a `MQRFH` header, all value entries in the name value pairs are copied to the corresponding `OUT` (or `INOUT`) parameter in the IDL file. The name has to match the part of the IDL parameter name after the prefix. In the above example consider that the MQ message has two name value pairs, `H1 v11` and `H3 v22`. Then the value `v11` will be assigned to the parameter `MQ_RFH_H1`, the parameter `MQ_RFH_H2` gets no value assigned, and the entry for `H3` will be ignored.

Character Encoding Issues

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

If the payload of the MQ message is in XML format (`property entirex.bridge.xmm` has been set), the WebSphere MQ RPC Server converts the XML payload to the encoding used for the remote procedure call. If the WebSphere MQ RPC Server 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 WebSphere MQ RPC Server converts the message to the encoding specified by the CCSID (Coded Character Set IDentification) of the queue manager. When receiving a message, the WebSphere MQ RPC Server 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 WebSphere MQ RPC Server.

User Exit for Message Processing

WebSphere 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 WebSphere MQ RPC Server 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 WebSphere 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 WebSphere MQ RPC Server 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 WebSphere MQ RPC Server.

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 MQ `GET`, the message is deleted from the queue immediately. If an error occurs in the further processing of the message within the WebSphere MQ RPC Server (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 MQ `PUT` operation 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 WebSphere MQ RPC Server 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 WebSphere MQ RPC Server.
2. The WebSphere MQ RPC Server 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 WebSphere MQ RPC Server 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 WebSphere MQ RPC Server.
2. The WebSphere MQ RPC Server 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 WebSphere MQ RPC Server creates a corresponding RPC reply which 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.

4 Administering the WebSphere MQ Listener

EntireX WebSphere MQ Listener runs as a listener on a WebSphere MQ queue and passes messages to an RPC server. It is used to send messages received from a WebSphere MQ queue to an RPC server applications. This means that existing RPC servers can be used for communication with WebSphere MQ.

The WebSphere MQ Listener can process MQ messages that are in XML or SOAP format and that can be mapped to a Software AG IDL file, using XML Mapping Editor. The resulting XMM file has to be specified by the configuration property `entirex.bridge.xmm`.

The WebSphere MQ Listener can also process MQ messages that are in text format. An IDL file is used, which describes the message layout (see ***Mapping IDL Data Types to the MQ Message Buffer***). The IDL file has to be specified by the configuration `property entirex.bridge.idl`. The program name is taken from the `applicationIdData` field of the incoming MQ message. If this field is empty and the IDL file has only one program this program will be called. A custom logic (e.g. using the first n bytes of the MQ message payload) can be implemented in the user exit.

Note: EntireX WebSphere MQ ignores EntireX configuration parameters inside the XML/SOAP payload. See *XML/SOAP Listener* under *Writing Advanced Applications with the XML/SOAP Wrapper*.

See also ***Introduction to the WebSphere MQ RPC Server and WebSphere MQ Listener*** | ***Advanced WebSphere MQ Listener Functionality***.

Customizing the EntireX WebSphere MQ Listener

To set up the WebSphere MQ Listener, there is a configuration file and there are scripts to start the WebSphere MQ Listener.

The WebSphere MQ Listener is contained in *entirex.jar*. There are two parts: the RPC side and the WebSphere MQ side.

The WebSphere MQ Listener uses the WebSphere MQ base Java classes from IBM. To run the WebSphere MQ Listener, you need either the base Java classes or a full installation of WebSphere. Prerequisites for all EntireX components are described centrally. See *Prerequisites for WebSphere MQ RPC Server and WebSphere MQ Listener* in the respective section of the Release Notes for the required JAR file(s). The WebSphere MQ environment variables `MQ_JAVA_LIB_PATH` and `MQ_JAVA_INSTALL_PATH` must be set.

Make sure that either the local WebSphere MQ installation or the WebSphere MQ Java classes are accessible.

The default name for the configuration file is *entirex.wmqbridgelistener.properties*. The WebSphere MQ Listener 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 WebSphere MQ Listener.

Alternatively, use script `wmqbridgelistener.bsh` (UNIX) or `wmqbridgelistener.bin` in the *bin* directory to start the WebSphere MQ Listener. Both scripts use the configuration file *entirex.wmqbridgelistener.properties* in the folder *etc*, and both can be customized.

Configuring the RPC Server Side

The WebSphere MQ Listener converts an MQ message into an RPC request to an RPC server. The RPC server is defined using the following properties.

Name	Default Value	Explanation
entirex.bridge.marshalling		Define the type of marshalling (Natural or COBOL). Must be set only if the IDL file contains arrays of groups. See *Mapping IDL Data Types to the MQ Message Buffer*.
entirex.server.brokerid	localhost	Broker ID. See *URL-style Broker ID*.
entirex.server.serveraddress	RPC/SRV1/CALLNAT	Server address
entirex.server.userid	WMQListener	The Broker user ID.
entirex.server.libname		The RPC library name (optional). The default value is the library name of the XMM/IDL file.
entirex.server.naturallogon	no	Enables or disables logon to Natural Security for Natural RPC Server.
entirex.server.reliableRPC	no	If set to "yes", use reliable RPC for the call to the RPC Server.
entirex.server.rpcuser		Optional. RPC user ID (if different from entirex.server.brokerid).
entirex.server.rpcpassword		Optional. RPC password (if different from entirex.server.password).
entirex.server.retrycycles	15	Number of retry attempts if the call to the RPC server is not successful. If all attempts fail, the MQ message will not be committed and the WebSphere MQ Listener will terminate. If a dead-letter queue has been specified, the message will be put to that queue and committed and the Listener will not stop.
entirex.server.retryinterval	20	Retry interval (in seconds) if the call to the RPC server is not successful.
entirex.server.compresslevel	0 (no compression)	Permitted values (you can enter the text or the numeric value): / BEST_COMPRESSION / 9 / BEST_SPEED / 1

Name	Default Value	Explanation
		DEFAULT_COMPRESSION: -1, mapped to 6 DEFLATED: 8 NO_COMPRESSION: 0 N: 0 Y: 8
entirex.server.encryptionlevel	0	Deprecated. For encrypted transport we strongly recommend using the Secure Sockets Layer/Transport Layer Security protocol. See *SSL/TLS and Certificates with EntireX*.
entirex.server.logfile		Name of the log file, the default is standard output.
entirex.server.monitorport	0	The port where the server listens for commands from the System Management Hub (SMH). If this port is 0, no port is used and management by the SMH is disabled.
entirex.server.password		The password for secured access to the Broker. The password is encrypted and written to the property entirex.server.password.e. To change the password, set the new password in the properties file (default is *entirex.wmqbridge.properties*). To disable password encryption, set entirex.server.passwordencrypt=no (default for this property is yes).
entirex.server.properties	entirex.wmqbridgelistener.properties	The file name of the property file.
entirex.server.security	no	no/yes/auto/Name of BrokerSecurity object.
entirex.server.logfile		Name of the log file, the default is standard output.
entirex.server.verbose	no	Enable verbose output to the log file.
entirex.server.waitserver	60S	Wait time for the call to the RPC server.
entirex.timeout	20	TCP/IP transport timeout. See *Setting the Transport Timeout* under *Writing Advanced Applications - EntireX Java ACI*.
entirex.trace	0	Trace level (1,2,3).

Configuring the WebSphere MQ Side

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

Name	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		Name of the (local or remote) queue manager. If not specified, a connection is made to the default queue manager.
entirex.wmqbridge.listenqueue		Name of the queue from which messages are retrieved.
entirex.wmqbridge.userid		User ID for MQ Server.
entirex.wmqbridge.password		Password for MQ Server.
entirex.wmqbridge.userexit		Class name for WMQBridge user exit.
entirex.wmqbridge.userexit.classpath		URL of the classpath for WMQBridge user exit (optional).
entirex.wmqbridge.ccsid	platform encoding	Coded Character Set Identification used by the WebSphere MQ Listener (which acts as an MQ client), unused in bindings mode.

Name	Default Value	Explanation
entirex.wmqbridge.mqtrace	0	MQ tracing enabled if parameter > 0.
entirex.bridge.idl		Name of a Software AG IDL file; messages to/from MQ are in plain text.
entirex.bridge.xmm		Name of XMM (XML mapping) file; messages to/from MQ will be converted to XML.
entirex.bridge.xml.encoding		Encoding of the reply XML document.
entirex.bridge.names.file		Name of a properties file generated with the *bridge.tpl* template, which contains names of the first level parameters in the IDL file (optional). Necessary if IDL file contains dynamic MQ parameters.
entirex.bridge.verbose	no	Verbose/trace mode of WebSphere MQ Listener
entirex.wmqbridge.environment.sslCipherSuite		Configuration for SSL connection to MQ Server. See the WebSphere MQ documentation for details.
entirex.wmqbridge.environment.sslFipsRequired		Configuration for SSL connection to MQ Server. See the WebSphere MQ documentation for details.
entirex.wmqbridge.priority		Message priority for messages sent to MQ (different from the default priority of the destination queue).
entirex.wmqbridge.deadletterqueue		Name of the queue that will receive

Name	Default Value	Explanation
		unprocessed messages.

Mapping IDL Data Types to the MQ Message Buffer

The WebSphere MQ Listener uses a predefined mapping of IDL data types to the MQ message buffer if the MQ message is in text format.

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

Data Type	Description	Format	Note
A*number*	Alphanumeric	*number* bytes, encoding the characters.	
AV	Alphanumeric variable length	Bytes up to the end of the buffer.	1, 4
AV[*number*]	Alphanumeric variable length with maximum length	Bytes up to the end of the buffer, maximal length *number*.	1
K*number*	Kanji	Same as data type A.	
KV	Kanji variable length	Same as data type AV.	1, 4
KV[*number*]	Kanji variable length with maximum length	Same as data type AV[*number*].	1
I1	Integer (small)	*sign* (+, -) and 3 bytes (digits).	
I2	Integer (medium)	*sign* (+, -) and 5 bytes (digits).	
I4	Integer (large)	*sign* (+, -) and 10 bytes (digits).	
N*number1*[.*number2*]	Unpacked decimal	*sign* (+, -), *number1* bytes (digits) [*number2*] bytes (digits), no decimal point.	
P*number1*[.*number2*]	Packed decimal	*sign* (+, -), *number1* bytes (digits) [*number2*] bytes (digits), no decimal point.	
L	Logical	1 byte: X for true, all other false.	
D	Date	*YYYYMMDD*.	2
T	Time	*YYYYMMDDhhmmssS*.	3

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.

Data types not supported:

- Binary (`B[n]`,`BV`, `BV[n]`)
- Floating point (`F4`, `F8`)

Starting the WebSphere MQ Listener

Use start script `wmqlistener.bsh` (UNIX) or `wmqlistener.bat` (Windows) in the folder *bin* to start the WebSphere MQ Listener. You may customize this file. See also *Prerequisites for WebSphere MQ RPC Server and WebSphere MQ Listener* in the respective section of the Release Notes.

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.

Stopping the WebSphere MQ Listener

Use CTRL-C to stop the WebSphere MQ Listener.

On UNIX you can use `kill <pid of java process>` to stop the WebSphere MQ Listener.

Using SSL/TLS

To use SSL with the WebSphere MQ Listener, you need to configure two sides:

- **WebSphere MQ Side**
  See parameters `entirex.wmqbridge.environment.sslCipherSuite` and `entirex.wmqbridge.environment.sslFipsRequired` under ***Configuring the WebSphere MQ Side***.
- **RPC Server side**
  RPC servers can use Secure Sockets Layer/Transport Layer Security (SSL/TLS) as the transport medium. The term “SSL” in this section refers to both SSL and TLS. RPC-based servers are always SSL clients. The SSL server can be either the EntireX Broker, Broker SSL Agent, or Direct RPC in webMethods Integration Server (IS inbound). For an introduction see SSL/TLS and Certificates in the Security 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 *Default Certificates Delivered with EntireX*.

2 Set up the WebSphere MQ RPC Server 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 EntireX Clients and Servers*.

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
- *Settting up and Administering the EntireX Broker SSL Agent* in the UNIX and Windows administration documentation
- *Support for SSL/TLS* in the EntireX Adapter documentation (for Direct RPC)

Tracing

The trace level for the EntireX RPC part is controlled by the usual `entirex.trace` property. It can be set in the configuration file. Additional diagnostic output can be enabled by setting the property `entirex.server.verbose`.

The WebSphere MQ Listener-specific diagnostic output is enabled by setting the property `entirex.bridge.verbose`. In addition, tracing of the IBM WebSphere MQ classes can be influenced with the property `entirex.wmqbridge.mqtrace`.

Use the RPC server agent of the System Management Hub to dynamically change the level of the diagnostic output. You can specify a value of 0, 1, 2, or 3 which changes the value of `entirex.trace`. In addition, the value 0 will disable both `entirex.server.verbose` and `entirex.bridge.verbose`. A value greater than 0 will enable both `entirex.server.verbose` and `entirex.bridge.verbose`.

Redirect the trace to a file with the property `entirex.server.logfile`. Set this to the file name of the log file.

5 Advanced WebSphere MQ Listener Functionality

Support for Synchronous Request/Reply Scenarios

The WebSphere MQ Listener provides support for a synchronous request/reply scenario. In this case the remote procedure call usually has both `INPUT` and `OUTPUT` parameters, or at least one `INOUT`. A request/reply scenario is automatically detected by the WebSphere MQ Listener if the MQ message specifies the `ReplyToQueue` field. Also, if the property `entirex.wmqbridge.check.msgtype` has been set, the message type must be "`MQMT_REQUEST`".

The `correlationId` of the reply is set to the `correlationId` of the request if the `report` field of the request has the flag `MQRO_PASS_CORREL_ID` set. Otherwise the `correlationId` of the reply is set to the `messageId` of the request.

The `messageId` of the reply is set to the `messageId` of the request if the `report` field of the request has the flag `MQRO_PASS_MSG_ID` set. Otherwise a new `messageId` is created.

Dynamic IDL/RPC Parameters for WebSphere MQ Listener

With the WebSphere MQ Listener it is possible that certain parameters of a remote procedure call are dynamic parameters which are evaluated by the RPC server. Dynamic parameters have a fixed name; they can be defined only on level 1 in the parameter definition in the IDL file and before any variable length parameter, and have a specific format. The WebSphere MQ Listener uses the following parameter. All other dynamic parameters described in the corresponding section of the WebSphere MQ RPC Server documentation are ignored.

Parameter	Format	Description
`MQRFH Header data (MQ_RFH_*)`	A with fixed length	Arbitrary number of parameters. The names following the prefix "`MQ_RFH_`" are used as the names of the name value pairs of the `MQRFH` header.

If dynamic parameters are to be used, generate a properties file from the corresponding Software AG IDL file and specified with the `entirex.bridge.names.file` property. To generate this property file, use the template *bridge.tpl* in the *template* subdirectory of the EntireX installation. For batch generation, run `erxidl.bat` (Windows) or `erxidl.bsh` (UNIX) with the parameters "`-t <path to template directory>/bridge.tpl <idlFile>`".

Alternatively, you can also use the *EntireX Workbench*. Go to the **Preferences for EntireX** and create a new Custom Wrapper. Specify a name and browse to the *bridge.tpl* template. If the Custom Wrapper has been created (and the Workbench restarted), you can generate the properties file from an IDL file, using the context menu item **Other > Generate...**.

Support for the MQRFH Header

The WebSphere MQ Listener supports the MQRFH header (rules and formatting header). This header consists basically of name value pairs. Restriction: only one header per MQ message is possible; MQ allows an arbitrary number of headers per message.

When sending a message to MQ: an MQRFH header is built if at least one parameter in the IDL file has a name with prefix "MQ_RFH_". All IN (or INOUT) parameters with this prefix are used to build the header. If for example the IDL file contains two fields `MQ_RFH_H1` and `MQ_RFH_H2` with the values "v1" and "v2", the resulting MQRFH header will have two name value pairs, "H1 v1" and "H2 v2".

If a message is received from MQ: if the message has an MQRFH header, all value entries in the name value pairs are copied to the corresponding OUT (or INOUT) parameter in the IDL file. The name has to match the part of the IDL parameter name after the prefix. In the above example consider that the MQ message has two name value pairs, "H1 v11" and "H3 v22". Then the value "v11" will be assigned to the parameter `MQ_RFH_H1`, the parameter `MQ_RFH_H2` gets no value assigned, and the entry for "H3" will be ignored.

Character Encoding Issues

If the payload of the MQ message is in XML format (property `entirex.bridge.xmm` has been set), the WebSphere MQ Listener converts the XML payload to an RPC request, using the default platform encoding of the Java virtual machine. If the WebSphere MQ Listener sends back a reply message, the XML payload of this message will be UTF-8 encoded. 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 (property `entirex.bridge.idl` has been set), the WebSphere MQ Listener converts the payload to an RPC request, using the default platform encoding of the Java virtual machine. If the WebSphere MQ Listener sends back a reply message, the Listener converts the payload of the message to the encoding specified by the CCSID (Coded Character Set IDentification) of the MQ queue manager.

Note: The default platform encoding of the JVM can be changed by setting the system property `file.encoding` in the startup script of the Listener.

User Exit for Message Processing

WebSphere 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 WebSphere MQ Listener 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 works on the WebSphere 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 read from an MQ queue and before it is processed by the WebSphere MQ Listener
2. after an MQ message has been constructed by the WebSphere MQ Listener and before the message is put to the MQ queue.

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

The WebSphere MQ Listener always works transactionally. A message is retrieved from the queue and then passed to the RPC server application. If no error occurs, the message is committed by the WebSphere MQ Listener. If the call to the RPC server is not successful, the call is retried as specified by properties `entirex.server.retrycycles` and `entirex.bridge.retryinterval`. If all retry attempts fail, the message is rolled back and the WebSphere MQ Listener terminates. If a dead-letter queue has been specified by the property `entirex.server.deadletterqueue`, the message will be put to that queue and committed, and the Listener will not stop.