SonicMQ Transport

This topic covers the SonicMQ Transport, which is implemented as an extension to the JMS Transport.

Sections include:

In addition to JMS Message types, SOAtest and certain Virtualize supporting tools and provisioning action tools support `MultiPartMessage` for the SonicMQ Transport. For JMS Message types, you need to use JMS Transport. For more information on the JMS Transport, see JMS.

For outbound messaging, a `MultiPartMessage` is constructed with a single part that wraps the request message as "text/xml" type. Users can specify the 'Content-ID' header field of the part via the Part Content ID field. For inbound messaging, both JMS and SonicMQ Transports can parse MultiPartMessage with multiple parts.

Creating New Tests for SonicMQ

To configure SOAtest to create tests for access SonicMQ, complete the following:

1. Complete the WSDL test creation wizard as normal (see Creating Tests From a WSDL for details).
2. Double-click the test node for the test that will be using SonicMQ.
3. In the right GUI panel, open the **Transport** tab and select **SonicMQ** from the Transport drop-down menu. Various options will display underneath the **Transport** drop-down menu:
    - Connection Settings
    - Queue/Topic
    - Messaging Model
    - Message Exchange Pattern
    - Message Type
    - Request Message Properties
    - Response Message Correlation
4. Configure the desired options as described in the following sections.

Configuring SonicMQ Options

After selecting **SonicMQ** from the Transport drop-down menu within the Transport tab of an appropriate tool, the following options display in the left pane of the **Transport** tab:

- Connection Settings
- Queue/Topic
- Messaging Model
- Message Exchange Pattern
- Message Type
- Part Content ID
- Request Message Properties
- Response Message Correlation

Connection Settings

Connection Settings contains the **Settings** and **Properties** tabs. The Properties tab is optional and allows users to specify additional properties to be passed to the JNDI javax.naming.InitalContext constructor; in addition to the Provider URL and Initial Context factory properties that are specified in the Settings tab. The Settings tab contains:

- If you created a Shared Property for SonicMQ Connections, a drop-down menu will be available from which you can choose **Use Local Settings** o r **Use Shared Property.**
    - If you select **Use Shared Property,** a second drop-down menu displays from which you select the desired global SonicMQ settings that the tool will use. For more information, see Adding Global Test Suite Properties

      .
    - If you select **Use Local Settings,** or if no shared property is specified, you can configure the rest of the options for Connection Settings.
- **Provider URL:** Specifies the value of the property named javax.naming.Context.PROVIDER_URL passed to the JNDI javax.naming.InitialContext constructor.
- **Initial Context:** Specifies a fully qualified class name string, passed to the JNDI javax.nam-ing.InitialContext constructor as a string value for the property named javax.naming.Context.INITIAL_CONTEXT_FACTORY.
- **Connection Factory:** Passed to the lookup() method in javax.naming.InitialContext to create a javax.jms.QueueConnectionFactory or a javax.jms. TopicConnectionFactory instance

In addition to the Settings tab, the Connection Settings also include:

- **Queue Connection Authentication:** Allows users to provide a username and password to create a queue connection. Select the **Perform Authentication** check box and enter the **Username** and **Password** to authenticate the request. If the correct username and password are not used, the request will not be authenticated.The username and password provided here is passed to the createQueueConnection() method in the javax.jms.QueueConnectionFactory class in order to get an instance of javax.jms.QueueConnection.

- **Keep-Alive Connection:** Select to notify the test whether to share or close the current connection. The shared connections are returned to the connection pool to be used across the test suite. A life cycle of a connection pool is as follows:
    - For a single test, it is destroyed at the end of the test execution.
    - For a test suite, it is destroyed at the end of the test suite execution

Queue/Topic

The Queue/Topic settings contain the following options:

- **JMS Destination:** Specifies the queue name (if point to point is used) or topic name (if publish and subscribe is used) for where the message will be sent to.
- **JMS ReplyTo:** Specifies the queue name (if point to point is used) or topic name (if publish and subscribe is used) for where to get a response message from. This can be a temporary queue if **Temporary** is selected instead of **Form.**

Messaging Model

Messaging Model options specify how messages are sent between applications. Select either **Point to Point** or **Publish and Subscribe**.

Message Exchange Pattern

Message Exchange Pattern options specify whether or not SOAtest or Virtualize receive a response. If **Get Response** is selected, SOAtest or Virtualize sends a message and receives a response. If **Get Response** is not selected, SOAtest or Virtualize sends a one-way message and does not receive a response.

If **Get Response** is selected, you can also enable **Create consumer on the JMSReplyTo destination before sending the message**.If the response is expected to become available very quickly on the JMSReplyTo topic, this option should be enabled to ensure that SOAtest or Virtualize has subscribed to the reply topic before the response message is published.

This option is cannot be mixed with **Match response JMSCorrelationID with the request JMSMEssageID** because the JMS specification requires vendors to generate the JMSMessageID after the message is sent. As a result, there is no way to create the consumer on the response destination with that correlation (selector) set until after the message has been set and the JMSMessageID becomes available.

Message Type

Message Type options allow you to select the message type from the drop-down menu. A SonicMQ Message is a Java object that contains the data being transferred between SonicMQ clients. The following Message Types are available:

- **progress.message.jclient.MultipartMessage**

Part Content ID

A Sonic MultipartMessage can have multiple parts. Each part has its own name (ID) and content. SOAtest and Virtualize support sending MultipartMessages with a single part. This field specifies the part name and the content is defined by the Request area, such as Form Input, Literal, Literal XML (SOAtest), etc. SOAtest and Virtualize support receiving MultipartMessages with multiple parts and outputs all the part contents as XML to the Response Output of the tool.

Request Message Properties

The Request Message Properties are optional and allows for any miscellaneous property values to be set into the javax.jms.Message object before it gets sent to a queue or published to a topic. These include predefined properties that get set to the outgoing requests message using one of the corresponding "set" methods in javax.jms.Message, or any custom property provided with the setStringProperty() method.

Response Message Correlation

The Response Message Correlation settings contain the following options:

- **Match response JMSCorrelationID with request JMSMessageID**: If selected, the term JMSCorrelationID = '[msgId]' will be appended to the selector expression, where msgId is dynamically generated from the outgoing (request) javax.jms.Message (using the getJMSMessageID() method). Effectively, this results in the tool blocking until a message with the specified correlation id becomes available in the queue (or topic) and it will only retrieve that particular message, rather than retrieving any message in the queue (or topic). The tool will timeout after the timeout amount elapses and if there is no message that watches the selector criteria.
- **Match response JMSCorrelationID with request JMSCorrelationID**: If selected, the term JMSCorrelationID = '[correlationId]' will be appended to the selector expression, where correlationId is retrieved from JMSCorrelationID property in the Message Properties section. The option becomes enabled only if such property is added to the Message Properties section. Effectively, this results in the tool blocking until a message with the specified correlation id becomes available in the queue (or topic) and it will only retrieve that particular message, rather than retrieve any

message in the queue (or topic). The tool will timeout after the timeout amount elapses and if there is no message that watches the selector criteria.

- **Additional Selector Expression Terms**: (Optional) Enter a value to act as a message filter. For tips on specifying a selector, see Using Message Selector Filters.

Message Object Outputs for Clients Using SonicMQ

You can add message object outputs to tools that utilize the SonicMQ transport. For example, an Extension tool chained to a tool that uses SonicMQ will have access to the response SonicMQ Message. In the ObjectMessage case, you can use `getter` and `equals()` methods to validate the response thereby creating a regression control. In addition, you can chain a Diff tool to the Response Traffic and if the response is an ObjectMessage, SOAtest or Virtualize will convert the inserted serializable object to XML format and perform an XML diff. By doing this you can use data bank values, ignore XPath differences, etc.

To do this complete the following:

1. Right-click SOAP Client or Messaging Client node, or the Virtualize tool node the node for which you would like to add the output and select **Add Output** from the shortcut menu. The Add Output wizard displays.
2. Select **Response> Message Object** in the left pane of the Add Output wizard and then choose a **New Output** or **Existing Output** and the desired tool (e.g. an Extension tool) from the right pane.
3. Click the **Finish** button. SOAtest or Virtualize adds the new output to the selected Client node.