Call Back

This topic explains how to configure and apply the Call Back tool that supports asynchronous HTTP testing by listening for the asynchronous server response.

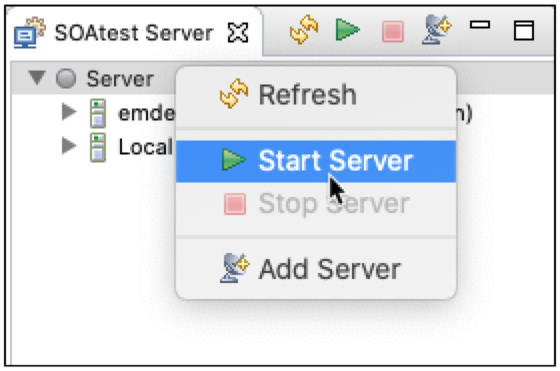
Sections include:

Understanding the Call Back Tool

SOAtest supports asynchronous HTTP testing, including Parlay, Parlay-X, SCP (SOAP Conversation Protocol), WS-Addressing, custom protocols, JMS, TIBCO, WebSphere MQ and SonicMQ. After you configure the Call Back tool, SOAtest sets up a server to manage the Call Back Messages. It uses the keys you specify in the tool configuration to recognize incoming messages.

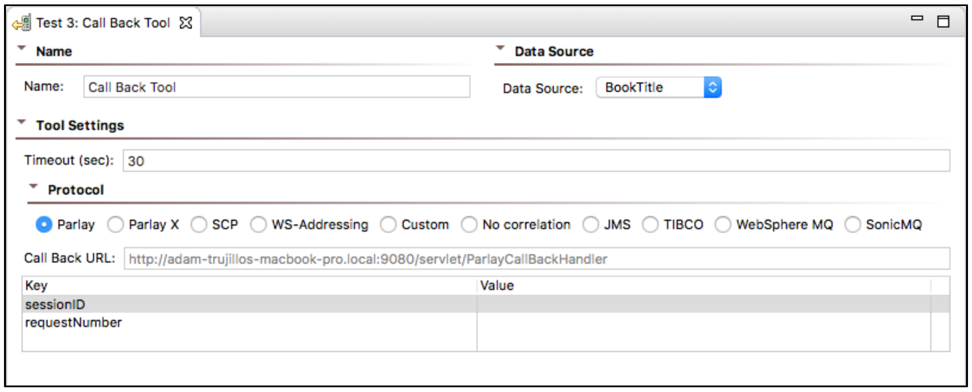
Configuring the Call Back Tool

Before you begin, verify that the local SOAtest server is running. The server can be started by right-clicking the Server node in the SOAtest Server view and choosing **Start Server**.



If the SOAtest Server view is not visible in your workspace, select **Parasoft> Show View> SOAtest Server.**

1. Ensure that there is a SOAP invocation from a SOAP Client tool.
2. Select the main test suite node and click the **Add test or output** button. The Add Test wizard displays.
3. Select **Standard Test> Call Back Tool** from the Add Test wizard, and then click the **Finish** button. A **Call Back Tool test** node displays in the test suite.
4. Double-click the **Call Back Tool** test node. The following options display in the Call Back tool workspace:



Test 3: Call Back Tool

Name

Name: Call Back Tool

Data Source

Data Source: BookTitle

Tool Settings

Timeout (sec): 30

Protocol

[✓ Parlay] Parlay X SCP WS-Addressing Custom No correlation JMS TIBCO WebSphere MQ SonicMQ

Call Back URL: http://adam-trujillos-macbook-pro.local:9080/servlet/ParlayCallBackHandler

Key	Value
sessionID	
requestNumber	

- **Data Source:** (Only available if a Data Source was added to the test suite) Select the data source that has the desired parameters to be used as call back values.
- **Call Back URL:** Displays the server location of the Call Back tool. *For HTTP protocols only*.
- **Timeout (sec):** Specifies the length of delay (in seconds) after which SOAtest should consider your requests to be timed out.
- **Protocol:** Specifies the asynchronous protocol to be used.
    - **Parlay:** For the Parlay protocol.
    - **Parlay X:** For the Parlay X protocol.
    - **SCP:** For the SCP protocol.
    - **WS-Addressing:** For the WS-Addressing protocol.

        - **Custom:** Enables custom message correlation over HTTP. The Call Back tool will pick up any messages at the call back URL that match the text value of elements at specified XPaths. For example, assume you have the following message:
          `<root>`
          `<id>2</id>`
          `<name>Java</name>`
          `</root>`
          You can define an XPath of `/root/id` with value 2. The XPath will select the id element. The Call Back tool will then check if the text content matches the configured value of 2. The Call Back tool will not pick up any received messages that have a different text value for the `id` element.
        - **No correlation:** Select this to use the first message in http://localhost:9080/servlet/NoCorrelationCallBackHandler. If this "protocol" is selected and there is no message at that location, the test will fail once it times out.
        - **JMS:** For the JMS protocol. If JMS is selected as the protocol, additional options will be made available. For more information, see Configuring JMS options for the Call Back Tool.
        - **TIBCO:** For the TIBCO protocol. For more information, see Configuring TIBCO options for the Call Back Tool.
        - **WebSphere MQ:** For the WebSphere MQ protocol.
        - **SonicMQ:** For the SonicMQ protocol.
    - **Key Set:** Displays Keys and Values used in the Call Back message. These options are not available if JMS was selected as the Protocol.
        - **Key:** The Keys will vary depending on the Protocol selected.
            - For **Parlay**, the keys used to determine a unique Call Back message are **sessionID** and **requestNumber**. For more information on Parlay specifications/API, see http://www.parlay.org.
            - For **Parlay X**, the key used to determine a unique Call Back message is **correlator**.
            - For **SCP**, the key used to determine a unique Call Back message is **conversationID**. For more information on SCP specifications/API, see http://dev2dev.bea.com/technologies/webservices/SOAPConversation.jsp.
            - For **WS-Addressing**, the key used to determine a unique Call Back message is **MessageID**. For more information on WS-Addressing specifications, see http://www-106.ibm.com/developerworks/library/specification/ws-add/
        - **XPath:** (Only available for Custom) Double-click to enter an XPath.
        - **Value:** (Not available for JMS) Double-click to enter either a Fixed value or a Parameterized value (if a Data Source is available).
5. Right-click the **Call Back Tool** test node and select **Add Output**. The Add Output wizard displays.
6. Select the desired tool from the **Add Output** wizard to create an output for the Call Back message from the server. For example, you can select the Diff tool to create a regression control on the Call Back message from the server.

Configuring TIBCO options for the Call Back Tool

If **TIBCO** is selected as the Protocol for the Call Back tool, a **TIBCO Properties** panel displays at the bottom of the tool configuration panel. SOAtest can listen on a local TIBCO DAEMON or a remote one. That is, the bus daemon could be running on the local machine or somewhere else. The following options are available:

- The **Timeout** field represents the amount of time SOAtest should block (wait) until a message becomes available on the specified **Reply Subject** value.
- **Daemon:** Specifies the server name or the server's IP followed by a colon (:) and the port number (e.g. 10.10.32.34:7500 or host_name:7500).
- **Network:** Specifies the network where the transport objects are located. The network parameter consists of up to three parts separated by a semicolon (;) in the form of `network`; `multicast groups`; `send address` (e.g. `lan0; 224.1.1.1; 244.1.1.6`). For more information, please refer to the Network Selection section of the TIBCO Rendezvous documentation.
- **Service:** Specifies TIBCO's service name.

TIBCO messages, when generated and accessed via one of the programming language APIs, can put content under named fields. Content (SOAP messages, XML, text, etc.) must be placed under a TIBCO message "field". To retrieve desired content from the message, the **Reply Field Name** must be provided. The field name is determined by the application that generates a TIBCO message, so to determine what that field is, one would need to know the field name that is used by the application (if unknown, the source code of the application that sends the TIBCO message should include the field name).

The **Message Delivery** field indicates what type of messages the Call Back tool should look for on the bus. This should correspond to the delivery type established by the message sender.

Additional information on the **Field Name** can be found in the TIBCO Rendezvous docs under **TIBCO Rendezvous Concepts> Fundamentals> Messages and Data> Fields**.

For more information on the **Reply subject** field and its meaning in TIBCO, refer to TIBCO docs under **TIBCO Rendezvous Concepts> Fundamentals> Names and Subject-Based Addressing**.

Configuring JMS options for the Call Back Tool

If **JMS** is selected as the Protocol for the Call Back tool, a **JMS Properties** panel displays at the bottom of the Call Back tool workspace. The following options are available:

Connection Settings

Connection Settings contains **Settings** and **Properties** tabs for **JNDI Initial Context**.

The **Properties** tab is optional and allows you 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). Property values, which can be added by clicking **Add** and completing the Add JMS Property dialog, can be set to a fixed value, a parameterized value, a scripted value, or a unique value (an automatically-generated random unique value—no two test invocations will use the same value).

The **Settings** tab contains the following:

- If you created a Shared Property for JMS Connections, a drop-down menu will be available from which you can choose **Use Local Settings** or **Use Shared Property**.
    - if you select **Use Shared Property**, a second drop-down menu displays from which you select the desired global JMS settings that the SOAP Client tool will use. For more information on global JMS settings, see Global JMS Connection 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.naming.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 or Topic Connection Authentication:** Allows you 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.

Queue/Topic

The Queue/Topic settings contain the following options:

- **JMS Address:** 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.

Messaging Model

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

Incoming Message Correlation

The Incoming Message Correlation settings contain the following options:

- When **Match incoming JMSCorrelationID with** is selected, the term JMSCorrelationID = '[correlationId]' will be appended to the selector expression, where correlationId is retrieved from the JMSCorrelationID property in the Message Properties section. The option becomes enabled only if such a property is added to the Message Properties section. Effectively, this results in the test 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 test will timeout after the timeout amount elapses and if there is no message that watches the selector criteria.
    - The criteria can be specified with a fixed value, a parameterized value (from a data source), or a scripted value.
- **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.