Monitoring IBM WebSphere ESB

This topic explains how to configure monitoring for IBM WebSphere ESB. Sections include:

WebSphere Configuration

IBM WebSphere ESB includes monitoring capabilities that build upon its underlying WebSphere Application Server. Parasoft SOAtest can subscribe to Common Base Events that are fired at points in the processing of service components, and which are managed by WebSphere Common Event Infrastructure (CEI).

For information about monitoring service component events in the WebSphere ESB and enabling the monitoring using WebSphere administrative console, see the IBM Monitoring service component events topic.

To configure WebSphere for event monitoring:

1. Enable the CEI service in the ESB.
2. Choose the level of logging for the service components you are interested in. The steps for performing this task on the ESB can be found at the IBM Configuring service component event monitoring using the administrative console topic.
    - In order to get the full event details in SOAtest, we recommend that you select the "ALL MESSAGES AND TRACES" option and the "FINEST" logging level for the components you are interested in, and which results in the business messages being included in the CEI events. To enable that for all business integration components, the log level string in the WebSphere administrative console would look like this:

      ```
      *=info: WBILocationMonitor.CEI.SCA.com.*=finest
      ```

SOAtest Configuration

Adding Required Jar Files to the SOAtest Classpath

The following jar files need to be added to the SOAtest classpath:

- com.ibm.ws.ejb.thinclient_7.0.0.jar
- com.ibm.ws.orb_7.0.0.jar
- com.ws.sib.client.thin.jms_7.0.0.jar
- com.ibm.ws.emf_2.1.0.jar

The jar files can be found under [WAS installation dir]/runtimes.

To add these jar files to SOAtest's classpath, complete the following:

1. Choose **Parasoft> Preferences**.
2. Open the **Parasoft> System Properties** page.
3. Click the **Add JARS** button and choose and select the necessary JAR files to be added.

Configuring the Event Monitor Tool

To configure the Event Monitor tool to monitor messages that pass through WebSphere ESB:

1. Double-click the Event Monitor tool to open up the tool configuration panel.
2. In the **Event Source** tab, select **IBM WebSphere Enterprise Service Bus** as the platform, then configure the following options:
    a. In the **Connection** area, specify your ESB connection settings.
        - The username and password are the credentials that were configured in the WebSphere ESB (under Security, Business Integration Security on the WebSphere administrative console for the Common Event Infrastructure).
        - The credentials you provide are used by SOAtest to create the JNDI InitialContext of the events JMS topic and to create the JMS connection.

    b. In the **Monitoring Source** field, specify the topic or queue that you want to monitor.
        - You can leave the default destination name as jms/cei/notification/AllEventsTopic, which is the CEI topic that reports all CEI events.
        - The connection URL is essentially the JNDI InitialContext URL for the WebSphere Default JMS provider.
        - The port number is the WebSphere bootstrap port.
        - You can check the correct port number for your WebSphere ESB using the administrative console under Servers section, WebSphere Application Server, then click or expand the "Ports" link under the "Communication" section. The port number to use in SOAtest is the BOOTSTRAP_ADDRESS value.

    c. (Optional) In the **Message Selector** field, enter a value to act as a message filter. See Using Message Selector Filters for tips.
    d. If you want SOAtest to use the JMS QueueBrowser API in order to trace messages posted on a JMS queue— without removing them from the queue— enable the **Leave messages on the queue** option. This allows SOAtest to gain visibility into these messages without impacting the transaction.

> **Caution: Leave messages on the queue**
>
> For a discussion of potential complications with this option—and how to avoid them—see JMS Queue Options.

e. If you want any additional JNDI properties applied to this deployment, specify them in the JNDI properties table.

3. In the **Options** tab, modify settings as needed.
   - **Clear the event viewer before each event monitor run** determines whether SOAtest automatically clears the Event Monitor event view (both text and graphical) when-ever Event Monitor starts monitoring.
   - **Include test execution events in the XML event output** specifies whether the Event Viewer tab and XML output display show only the monitored messages and events, or if they also indicate when each test started and completed. Enabling this option is helpful if you have multiple tests in the test suite and you want to better identify the events and correlate them to your test executions.
   - **Wrap monitored messages with CDATA to ensure well-formedness of the XML event output** should be disabled if you expect the monitored events' message con-tent to be well-formed XML. This will make the messages inside the events accessible via XPaths, allowing the message contents to be extracted by XML Transformer or validated with XML Assertor tools.
     - If the message contents are not necessarily XML, you should enable this option to ensure that the XML output of the Event Monitor tool (i.e. the XML Event Output for chaining tools to the Event Monitor, not what is shown under the Event Viewer) is well-formed XML by escaping all the message contents. This will make the content of these messages inaccessible by XPath since the message technically becomes just string content for the parent element.
     - Note that the Diff tool's XML mode supports string content that is XML. In other words, no matter which option you select here, the Diff tool will still be able to diff the messages as XML, including the ability to use XPaths for ignoring values.
   - **Maximum monitor execution duration** specifies the point at which the test should timeout—in case another test in the test suite hangs, or if no other tests are being run (e.g., if you execute the Event Monitor test apart from the test suite, then use a custom application to send messages to system).
   - **Event polling delay after each test finishes execution (milliseconds)** is not applicable here.

Viewing Monitored Events

After the test runs, the Event Monitor will show the XML representation of the Common Base Events it receives from WebSphere, including the event's raw business data if it is present.