

Muse 2.x Conformance Testing Tools ([MUSE-260](#))

Feature Description/Requirements

The Muse 2.x Conformance Testing Tools will be used to verify the conformance of resource endpoints with standard Muse capabilities and/or user custom capabilities to the specification versions implemented in Muse 2.x (WSRF 1.2, WSN 1.3, WSDM-MUWS 1.1, WSDM Event Format 1.1, and WS-MetadataExchange).

Design and Architecture

The WS-I Organization (<http://ws-i.org>) has developed and made available the WS-I Interoperability Testing Tools to help developers determine whether their Web Services are conformant with defined WS-I profiles.

These tools test Web service implementations using a non-intrusive, black box approach. We will extend and enhance these testing tools with additional assertion statements and associated code logic to test for conformance against the specification levels implemented in Muse 2.x.

More details on these testing tools and how they work can be found at <http://ws-i.org/deliverables/workinggroup.aspx?wg=testingtools> (under "[Analyzer Tool Functional Specification](#)", "[Monitor Tool Functional Specification](#)", and "Interoperability Testing Tools 1.1") and <http://www-128.ibm.com/developerworks/webservices/library/ws-wsitest/>.

As depicted in figure 1, the testing infrastructure is made up of two main modules, the Monitor and the Analyzer.

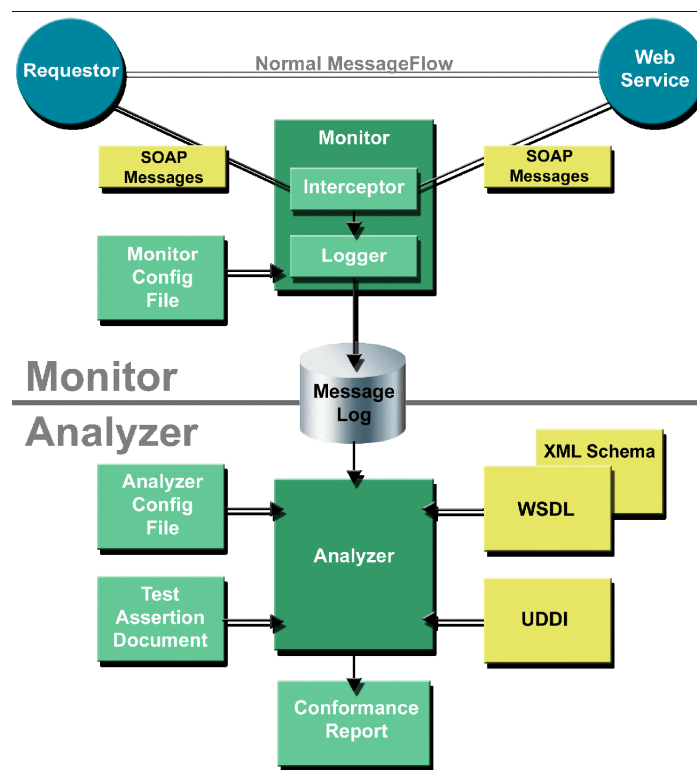


Figure 1. Testing Tools Architecture.

The Monitor is both a message capture and logging tool. It intercepts the messages and a logger re-formats them and stores them in a message log for later analysis. The monitor is implemented using a man-in-the-middle approach. It intercepts and records the messages, then it forwards to the messages to its final destination. No changes or enhancements will be needed for the Monitor component.

The Analyzer is an analysis tool that verifies the conformance of Web Services artifacts to profiles/specifications. It analyzes the messages sent to and from a Web service after these have been captured and stored in the message log by the Monitor. The Analyzer performs its checks based on assertions listed in a Test Assertion Document (TAD). A new TAD will be created with assertion statements extracted from the specifications implemented in Muse 2.x. The associated code logic that the Analyzer will invoke will also be added to this component. These TAD assertion statements will be listed in a separate, soon-to-follow document or in a later version of this same document under the Test Scenarios section.

Usage/Execution Scenarios

As the default, we will create and use an endpoint service which exposes all the operations implemented by the default capabilities in Muse. An endpoint client will also be created to invoke and exercise the operations and flows exposed by the endpoint service. The endpoint and corresponding client code will be generated from a WSDL file.

The conformance checking will be performed by executing the following general steps:

1. Deploy and start the resource endpoint (on the appropriate container)
2. Start the Monitor
3. Exercise the resource endpoint operations by running an endpoint client
4. Stop the Monitor
5. Run the Analyzer
6. View the conformance report generated by the Analyzer

Test Scenarios

(Refer to the individual, separate documents for the conformance tests for each of the individual specifications.)