Basic Profile Testing Methodology Working Group Approval Draft

Document Type:

Test Methodology Description, Working Group Approval Draft for Basic Profile, 4/28/2009

Editors:

Ram Jeyaraman, Microsoft
Jacques Durand, Fujitsu
Monica J. Martin, Microsoft

Last Edit Date:

6/1/2009

Document Status:

Version 1.0

This document is a Working Group Draft; it has been accepted by the Working Group as reflecting the current state of discussions. It is a work in progress, and should not be considered authoritative or final; other documents may supersede this document.

Notice

The material contained herein is not a license, either expressly or impliedly, to any intellectual property owned or controlled by any of the authors or developers of this material or WS-I. The material contained herein is provided on an "AS IS" basis and to the maximum extent permitted by applicable law, this material is provided AS IS AND WITH ALL FAULTS, and the authors and developers of this material and WS-I hereby disclaim all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, and of lack of negligence. ALSO, THERE IS NO WARRANTY OR CONDITION OF TITLE, QUIET ENJOYMENT, QUIET POSSESSION, CORRESPONDENCE TO DESCRIPTION OR NON-INFRINGEMENT WITH REGARD TO THIS MATERIAL.

IN NO EVENT WILL ANY AUTHOR OR DEVELOPER OF THIS MATERIAL OR WS-I BE LIABLE TO ANY OTHER PARTY FOR THE COST OF PROCURING SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY WAY OUT OF THIS OR ANY OTHER AGREEMENT RELATING TO THIS MATERIAL, WHETHER OR NOT SUCH PARTY HAD ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

Feedback

The Web Services-Interoperability Organization (WS-I) would like to receive input, suggestions and other feedback ("Feedback") on this work from a wide variety of industry participants to improve its quality over time.

By sending email, or otherwise communicating with WS-I, you (on behalf of yourself if you are an individual, and your company if you are providing Feedback on behalf of the company) will be deemed to have granted to WS-I, the members of WS-I, and other parties that have access to your Feedback, a non-exclusive, non-transferable, worldwide, perpetual, irrevocable, royalty-free license to use, disclose, copy, license, modify, sublicense or otherwise distribute and exploit in any manner whatsoever the Feedback you provide regarding the work. You acknowledge that you have no expectation of confidentiality with respect to any Feedback you provide. You represent and warrant that you have rights to provide this Feedback, and if you are providing Feedback on behalf of a company, you represent and warrant that you have the rights to provide Feedback on behalf of your company. You also acknowledge that WS-I is not required to review, discuss, use, consider or in any way incorporate your Feedback into future versions of its work. If WS-I does incorporate some or all of your Feedback in a future version of the work, it may, but is not obligated to include your name (or, if you are identified as acting on behalf of your company, the name of your company) on a list of contributors to the work. If the foregoing is not acceptable to you and any company on whose behalf you are acting, please do not provide any Feedback.

Feedback on this document should be directed to wsi-test-comments@ws-i.org.

Table of Contents:

1 Overview 3
2 Test Assertions 6
3 Test Scenarios 7
4 Test Log Files 8
5 Test Analysis 9
5.1 The Test Analyzer 9
5.2 The Test Report 10
6 Appendix 12
6.1 Appendix A: Test Assertion Structure 12
6.2 Appendix B: Sample Test Log File 14
6.3 Appendix C: Q & A on What to Expect from the Testing Tools 33

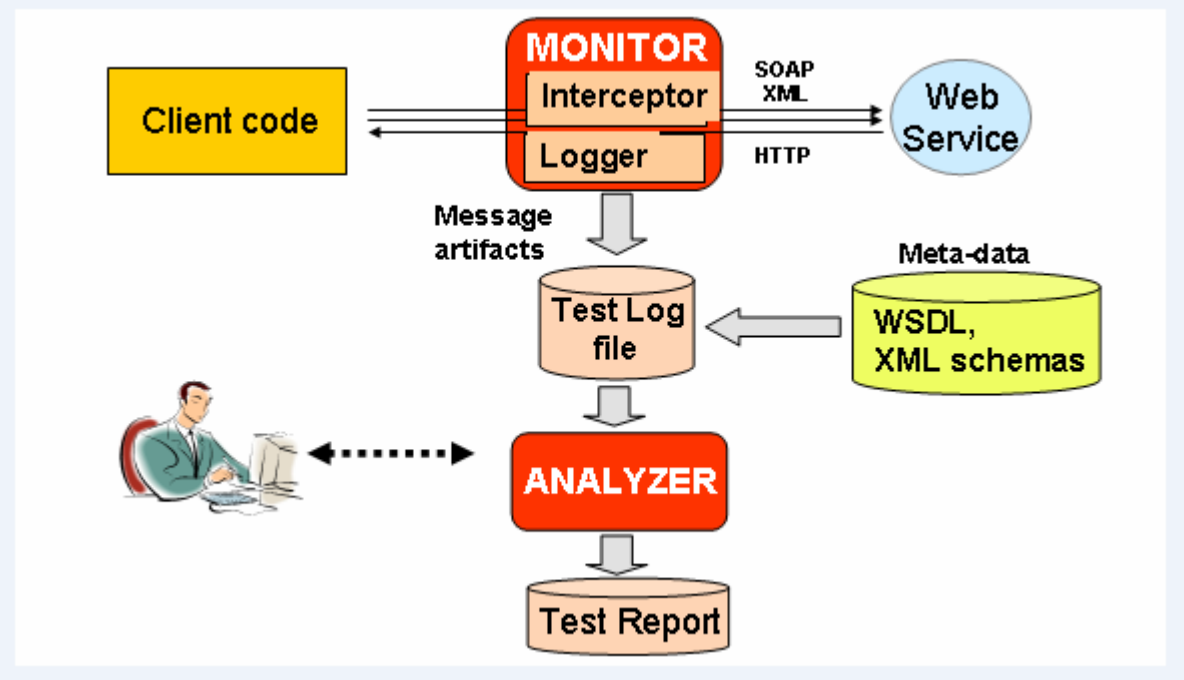
1 Overview

The testing approach used for the latest WS-I profiles (Basic Profile 1.2 and 2.0, Reliable and Secure Profile 1.0) differs from the one used in the past, although overall the general testing process remains similar: a "logging" phase including live message capture, followed by a distinct Analysis phase that involves a separate tool.

The overall test process involves two phases:

- Test log creation, including message capture and metadata consolidation (WSDL definitions, etc.)
- Test analysis, including evaluation of WS artifacts in the test log, against the requirements of a particular profile

The following figure illustrates the test process. A difference with the previous approach is that the Test Log file now consolidates all artifacts under test (messages + metadata).



The new test methodology uses the following artifacts for testing:

1. Enhanced test assertions embedded in the profile itself. Note: This replaces the Test Assertion document (TAD).

a. the enhanced test assertions pertain to requirements in the profile. The test assertions are composed of XPATH 2.0 expressions and hence language independent.

b. The profile itself encompasses the enhanced test assertions that replace the previous TAD. The test assertions are described as part of each profile requirement.

c. This is currently being developed by the BP WG. The enhanced test assertions describe the test assertion logic associated with exercising the applicable requirement(s). They are published by WS-I as a part of the applicable profile (see Basic Profile).

2. Test scenarios

a. The test scenarios are implemented and deployed on Web services runtime environment provided by various vendor platforms.

b. This exercises the various requirements in the profile.

c. This is currently being developed by the BP WG. The current scenario packages are published by WS-I (see Basic Profile).

3. Message logs

a. A monitoring method or tool is used to capture XML wire messages, and where required, a transformation tool is then used to convert them into a message log format as defined by the schema of the log message file format. These transformed message logs are then input into the test analyzer.

b. The log message file format is being provided as a part of the analyzer package (See Basic Profile).

4. Test analyzer

a. This takes the captured XML messages and test assertions (embedded in the profile) as input and produces an HTML test report.

b. The test analyzer is part of an analyzer package that includes the Test Assertion enhanced profile, and other components to be used to test the profile (see Message Logs).

Following are a set of steps to test a Web services application:

1. Capture message logs generated by the application.

2. Convert them to message log format (conforming to the schema for message log file format and using the transform).

3. Feed the converted message logs to the test analyzer and produce a HTML test report.

2 Test Assertions

A test assertion is a translation of a WS-I profile requirement into a statement verifiable by the analyzer. The following differences must be noted compared with the previous WS-I testing approach:

Previous approach: Test Assertions associated with a profile are defined in a separate XML document called the Test Assertion Definition doc (TAD).
New approach: Test Assertions associated with a profile are defined inside the Profile document itself, as included XML elements that are closely associated with the Profile requirement they address.

Previous approach: Test Assertions are defined in plain English in their host document, while their executable version is coded (C# and Java) inside the Analyzer tool.
New approach: Test Assertion elements are now scripted using XPath2.0. The Analyzer tool is able to directly process them. The XPath expressions are also visible to the end-user from the HTML rendering of the Profile document.

The structure of Test Assertions is still overall similar to those used in the previous test approach. Some of its elements have been renamed, and some added. A more detailed description is provided in Appendix A.

3 Test Scenarios

The test scenarios for Basic Profile exercise the requirements held in the profile. Participating partners exchange messages to test interoperability using SOAP, WSDL, MTOM and WS-Addressing. Tests leverage different SOAP binding styles and uses as well as existing test suites for underlying specifications.

Testing some of the requirements specified in Basic Profile can be easily done by examining common message flows between two endpoints. However, in some cases the requirements might require certain pre-conditions or runtime semantics that necessitate a client or a service endpoint to take certain actions that are specific to the tests defined by the Basic Profile test scenario documentation. In such cases, implementations being tested need to determine how best to recreate those environmental conditions; for example, a specialized test driver may be needed to implement some parts of the test scenarios.

The test scenarios are published by WS-I (see Basic Profile).

4 Test Log Files

Test Log files are provided as input to the test analysis phase (Analyzer method or tool), and are produced by the Monitoring phase involving run-time execution of test scenarios and the resulting message capture.

An enhanced version compatible with the former “Monitor log file” (the test input file for the Analyzer) has been defined and used by the new test approach. The test input file, also called the “Test Log file” is consolidating test artifacts as diverse as:

- service definition files (WSDL, Schemas)
- HTTP and MIME message headers
- SOAP envelopes and content

These artifacts are published by WS-I (see Basic Profile).

5 Test Analysis

5.1 The Test Analyzer

Unlike the previous test approach, where the Analyzer tool was developed in conventional programming languages (C#, Java) the new Analyzer is developed using XML processing technologies that are portable on any major platform, and based on W3C standards.

The Analyzer core module has been developed using XSLT 2.0. It can be run using any XSLT2.0 processor, although some advanced features used in some Test Assertions (TAs) (like schema validation) may not be available in every offering on the market for which a sufficient offering exists today for both .NET and Java platforms - including freeware such as products from Saxonica (Saxon-B or Saxon-SA) and AltovaXML.

The test analysis phase works in two steps. The first step only needs be executed once for each new profile:

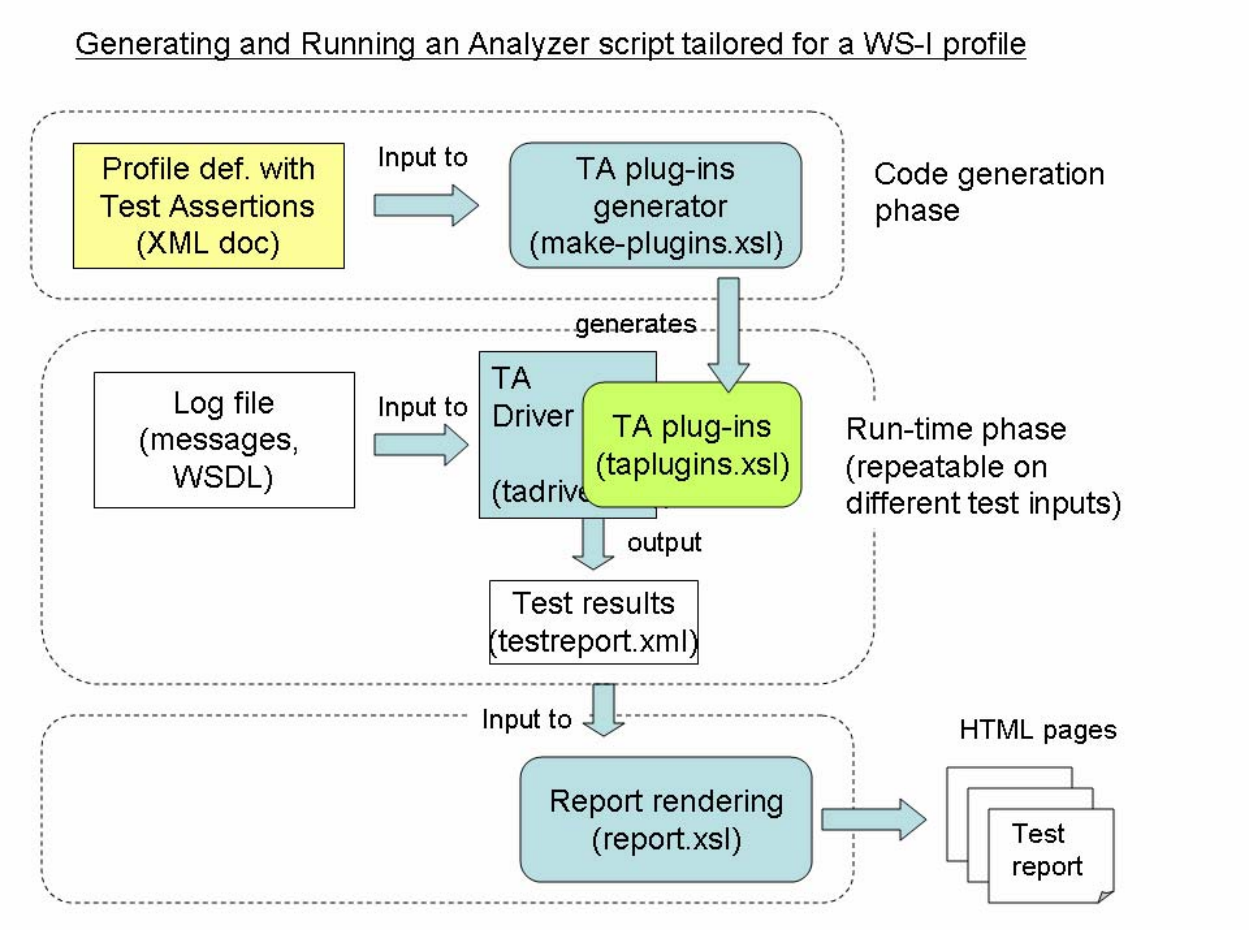
Step 1: Generation of the Analyzer code for a particular profile.

Input: the Profile definition document (XML) with embedded TA definitions.
Output: an XSLT script that represents the actual Analyzer tool for this profile.
Processor: a code generator written in XSLT2.0.

Step 2: Run-time analysis of Web services artifacts.

Input: a test log file.
Output: a test Report file in the same format as the previous generation of test tools V1.0, so that the same HTML rendering XSLT script can be reused.
Processor: the Analyzer XSLT tool produced by Step 1 for this profile.

These two steps are illustrated in the following figure:



- The "blue boxes" in the above figure, represent code that is permanent and does not change from one profile to the other.
- The "yellow/green boxes" represent code or data that varies from one profile to the other.
- The "white boxes" represent data that changes from one test run to the other.

5.2 The Test Report

The test report resulting from the processing of a test log file (Step 2) is an HTML document. It provides, for each artifact of the log file that is a target of at least one Test Assertion, a detailed report about this target. In particular, every TA that applies to this target will have its result (see list that follows) listed for the target, along with error messages when appropriate.

A cross-summary is also provided: for each Test Assertion, how many targets returned each one of the possible results shown in the list that follows.

When a test assertion is processed over a target, it will complete with one of the following outcomes:

- **passed**
  The test assertion script completed the verification on the target without detecting any errors. This is an indicator of conformance to the profile requirement.
- **failed**
  The test assertion script detected some error indicating that the target does not conform to the profile requirement.
- **warning**
  The test assertion was not conclusive about the target. The user is advised to more closely examine the target material or focus future testing on this test assertion as there is a potential for conformance failure.
- **notRelevant**
  The target did not qualify for this test assertion, which means that although the target was of the type of artifact relevant to this test assertion, it failed the prerequisite expression or it failed a prerequisite test assertion. In both cases, the test assertion is not relevant to this target.
- **missingInput**
  The test assertion was not processed due to missing collateral input (see "cotarget" in Test Assertion structure in Appendix).
- **notApplicable**

  The test assertion is not exercised for this target. The test assertion may not be enabled, or the target type of the test assertion does not even match this target. This result is usually not reported for any particular target, but will indicate that a test assertion was not exercised at all during an analysis run.

6 Appendix

6.1 Appendix A: Test Assertion Structure

The structure of a Test Assertion in the new test approach is defined as follows. This structure maps to an XML mark-up.

- **Description**: this is a plain English statement of the Test Assertion.
- **TA Id**: the identifier of the Test Assertion.
- **Source**: the normative profile requirement that this test assertion is addressing, identified by its requirement ID (Rxxxx).
- **Target:** (formerly "primary Entry Type") A Test Assertion always targets instances of a specific artifact type, for example, a SOAP Envelope, or a message (itself containing a SOAP envelope) or a WSDL port binding, etc. The Target element identifies this artifact type. The Target contains an XPath expression that defines the set of instances to which the TA applies, when evaluated over an XML file containing message capture and metadata (see Test Log file). The set of instances to be considered can be further narrowed by conditions (XPath predicates) inside the XPath expression. The Target expression was called the "Context" in previous test approach.
- **Cotarget:** (formerly "secondary Entry Type") A pointer to another artifact that is required to evaluate the TA over the specified Target. It is usually correlated with the Target in some way. For example, the Target may be a request message as captured on the wire, and the Cotarget may be the message parts description in WSDL that relates to this wire message. The Cotarget is scripted as an XPath expression that will select the related material in the Test Log file.
- **Pre-requisite:** A pre-condition that must be satisfied over the Target instance (and possibly its cotarget) in order for this Target to qualify for this TA. The Pre-requisite may refer to other Test Assertion that must be passed by the Target in order to qualify. If the Pre-requisite evaluates to "false", then the outcome of the TA for this Target will be "notRelevant" in the test report.
- **Predicate**: (formerly the "Assertion") A logical expression that can be evaluated over the Target (and Cotarget if appropriate). The Predicate is only evaluated if the TA is applicable, i.e. if the Pre-requisite – if any – has already evaluated to "true". In general, if the Predicate result is "true" then the Target fulfills the profile requirement addressed by this TA. If the result is "false" then the Target violates the profile requirement. However the predicate may be worded in a different way, e.g. to support negative testing. The interpretation of its result in the final test report is controlled by the *Reporting* element.

- **Prescription level**: Indicates the level of compliance expected from the Target, to the profile requirement. Possible values: mandatory, preferred, permitted. These values match the RFC2119 keywords MUST, SHOULD and MAY, and their equivalent.
- **Reporting**: Indicates how to interpret the Predicate outcome (true/false) from a conformance point of view in the test report. Possible reporting results are: “passed”, “failed”, “warning”, “undetermined”. In an ideal situation, the reporting map (which is also the default mapping) is: true=”passed”, false=”failed”. Due to testing constraints, the Predicate may not provide as good an indicator as the default one. It can then be mapped as: true=”warning”, false=”failed” or in other ways.

6.2 Appendix B: Sample Test Log File

An example is given in Figure 3 below:

```
<?xml version="1.0" encoding="UTF-8" ?>
wsil:testLog xmlns:wsil="http://www.ws-i.org/testing/2008/02/log/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
 xsi:schemaLocation="http://www.ws-i.org/testing/2008/02/log/ log.xsd">
wsil:logMetadata />
wsil:messageLog>
wsil:message type="request" conversation="1" id="1">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
 </wsil:contentTypeHeader>
 </wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:a="http://www.w3.org/2005/08/addressing"
 xmlns:xyz="http://schemas.myspace.org/">
s:Header>
a:To s:mustUnderstand="1">http://localhost:54564/SupplierService.svc</a:To>
a:Action
 s:mustUnderstand="1">http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncReceiveDelivery</a:Actio
a:MessageID>urn:uuid:9e233512-1e9d-4520-8bef-811a849085bc</a:MessageID>
a:ReplyTo>
a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
 </a:ReplyTo>
a:RelatesTo>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</a:RelatesTo>
xyz:other>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</xyz:other>
 </s:Header>
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
SyncReceiveDelivery systemEnvironmentCode="Test" languageCode="US-en" releaseID="9.0" versionID="2.0a"
 xmlns="http://www.openapplications.org/oagis/9/aiag/2">"test data"</SyncReceiveDelivery>
```

```
</s:Body>
</s:Envelope>
</wsil:messageContents>
</wsil:message>
wsil:message type="request" conversation="2" id="1">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
</wsil:contentTypeHeader>
</wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:xyz="http://schemas.myspace.org/">
s:Header>
a:To s:mustUnderstand="1">http://localhost:54564/SupplierService.svc</a:To>
a:Action
 s:mustUnderstand="1">http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncReceiveDelivery</a:Actio
a:MessageID>urn:uuid:9e233512-1e9d-4520-8bef-811a849085bc</a:MessageID>
a:ReplyTo>
a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
</a:ReplyTo>
a:RelatesTo>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</a:RelatesTo>
xyz:routinginput role="http://www.w3.org/2003/05/soap-envelope/role/next" a:IsReferenceParameter="true" />
</s:Header>
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
SyncReceiveDelivery systemEnvironmentCode="Test" languageCode="US-en" releaseID="9.0" versionID="2.0a"
 xmlns="http://www.openapplications.org/oagis/9/aiag/2">"test data"</SyncReceiveDelivery>
</s:Body>
</s:Envelope>
</wsil:messageContents>
</wsil:message>
wsil:message type="request" conversation="3" id="1">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
```

```
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
 </wsil:contentTypeHeader>
 </wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:xyz="http://schemas.myspace.org/">
s:Header>
a:To s:mustUnderstand="1">http://localhost:54564/SupplierService.svc</a:To>
a:Action
 s:mustUnderstand="1">http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncReceiveDelivery</a:Actio
a:MessageID>urn:uuid:9e233512-1e9d-4520-8bef-811a849085bc</a:MessageID>
a:ReplyTo>
a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
 </a:ReplyTo>
a:RelatesTo>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</a:RelatesTo>
xyz:routinginput role="http://www.w3.org/2003/05/soap-envelope/role/new" a:IsReferenceParameter="true" />
 </s:Header>
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
SyncShipment systemEnvironmentCode="Test" languageCode="US-en" releaseID="9.0" versionID="2.0a"
 xmlns="http://www.openapplications.org/oagis/9/aiag/2">
ApplicationArea>
Sender>
LogicalID schemeAgencyID="16" xmlns="http://www.openapplications.org/oagis/9">F159B</LogicalID>
ConfirmationCode xmlns="http://www.openapplications.org/oagis/9">OnError</ConfirmationCode>
CommonAccessReference>MinMax</CommonAccessReference>
 </Sender>
CreationDateTime xmlns="http://www.openapplications.org/oagis/9">2007-06-27T15:48:55Z</CreationDateTime>
BODID xmlns="http://www.openapplications.org/oagis/9">cbb44a11-4f95-4d1f-b489-bf93f24817c6</BODID>
Receiver>
LogicalID schemeAgencyID="16" xmlns="http://www.openapplications.org/oagis/9">F650A</LogicalID>
 </Receiver>
 </ApplicationArea>
DataArea nil="false">
Sync xmlns="http://www.openapplications.org/oagis/9">
```

```
ActionCriteria>
ActionExpression actionCode="Update" expressionLanguage="XPATH">aiag:MINMAX</ActionExpression>
 </ActionCriteria>
 </Sync>
ReceiveDelivery>
ReceiveDeliveryItem xmlns="http://www.openapplications.org/oagis/9">
CustomerItemID>
ID>1234</ID>
 </CustomerItemID>
DocumentReference type="123">
DocumentID>
ID>b5c8329f-2b17-4e1f-888b-bd9d10d248c6</ID>
 </DocumentID>
 </DocumentReference>
ReceivedQuantity>2</ReceivedQuantity>
 </ReceiveDeliveryItem>
 </ReceiveDelivery>
 </DataArea>
 </SyncShipment>
 </s:Body>
 </s:Envelope>
 </wsil:messageContents>
 </wsil:message>
wsil:message type="request" conversation="4" id="1">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
 </wsil:contentTypeHeader>
 </wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:a="http://www.w3.org/2005/08/addressing">
s:Header>
a:To s:mustUnderstand="1">http://localhost:54564/SupplierService.svc</a:To>
```

```
a:Action
 s:mustUnderstand="1">http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncReceiveDelivery</a:Actio
a:MessageID>urn:uuid:9e233512-1e9d-4520-8bef-811a849085bc</a:MessageID>
a:ReplyTo>
a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
 </a:ReplyTo>
a:RelatesTo>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</a:RelatesTo>
 </s:Header>
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
SyncQuantityOnHead systemEnvironmentCode="Test" languageCode="US-en" releaseID="9.0" versionID="2.0a"
 xmlns="http://www.openapplications.org/oagis/9/aiag/2">
SQOH1>
Sender>
LogicalID schemeAgencyID="16" xmlns="http://www.openapplications.org/oagis/9">F159B</LogicalID>
ConfirmationCode xmlns="http://www.openapplications.org/oagis/9">OnError</ConfirmationCode>
CommonAccessReference>MinMax</CommonAccessReference>
 </Sender>
 </SQOH1>
SQOH2>
Sender>
LogicalID schemeAgencyID="16" xmlns="http://www.openapplications.org/oagis/9">F159B</LogicalID>
ConfirmationCode xmlns="http://www.openapplications.org/oagis/9">OnError</ConfirmationCode>
CommonAccessReference>MinMax</CommonAccessReference>
 </Sender>
 </SQOH2>
SQOH3>
Sender>
LogicalID schemeAgencyID="16" xmlns="http://www.openapplications.org/oagis/9">F159B</LogicalID>
ConfirmationCode xmlns="http://www.openapplications.org/oagis/9">OnError</ConfirmationCode>
CommonAccessReference>MinMax</CommonAccessReference>
 </Sender>
 </SQOH3>
 </SyncQuantityOnHead>
 </s:Body>
 </s:Envelope>
```

```
</wsil:messageContents>
</wsil:message>
wsil:message type="request" conversation="25" id="1">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
</wsil:contentTypeHeader>
</wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:a="http://www.w3.org/2005/08/addressing">
s:Header>
a:To s:mustUnderstand="1">http://localhost:54564/SupplierService.svc</a:To>
a:Action
 s:mustUnderstand="1">http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncQuantityOnFoot</a:Actio
a:MessageID>urn:uuid:9e233512-1e9d-4520-8bef-811a849085bc</a:MessageID>
a:ReplyTo>
a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
</a:ReplyTo>
a:RelatesTo>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</a:RelatesTo>
</s:Header>
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
SyncQuantityOnFoot systemEnvironmentCode="Test" languageCode="US-en" releaseID="9.0" versionID="2.0a"
 xmlns="http://www.openapplications.org/oagis/9/aiag/2">
SQOH1>
Sender>
LogicalID schemeAgencyID="16" xmlns="http://www.openapplications.org/oagis/9">F159B</LogicalID>
ConfirmationCode xmlns="http://www.openapplications.org/oagis/9">OnError</ConfirmationCode>
CommonAccessReference>MinMax</CommonAccessReference>
</Sender>
</SQOH1>
</SyncQuantityOnFoot>
</s:Body>
</s:Envelope>
```

```
</wsil:messageContents>
</wsil:message>
wsil:message type="request" conversation="26" id="1">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
</wsil:contentTypeHeader>
</wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:a="http://www.w3.org/2005/08/addressing">
s:Header>
a:To s:mustUnderstand="1">http://localhost:54564/SupplierService.svc</a:To>
a:Action
 s:mustUnderstand="1">http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncQuantityOnFoot</a:Actio
a:MessageID>urn:uuid:9e233512-1e9d-4520-8bef-811a849085bc</a:MessageID>
a:ReplyTo>
a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
</a:ReplyTo>
a:RelatesTo>urn:uuid:0cb90750-c4ba-4ec7-bcaf-028c7ec4c842</a:RelatesTo>
</s:Header>
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" />
</s:Envelope>
</wsil:messageContents>
</wsil:message>
wsil:message conversation="26" id="2" type="response">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
</wsil:contentTypeHeader>
</wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
 xmlns:a="http://www.w3.org/2005/08/addressing">
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
s:Fault xmlns:m="http://www.example.org/timeouts">
s:Code>
s:Value>s:Sender</s:Value>
s:Subcode>
s:Value>m:MessageTimeout</s:Value>
 </s:Subcode>
 </s:Code>
s:Reason>
s:Text xml:lang="en">Sender Timeout</s:Text>
 </s:Reason>
 </s:Fault>
 </s:Body>
 </s:Envelope>
 </wsil:messageContents>
 </wsil:message>
wsil:message type="response" conversation="27" id="2">
wsil:httpHeaders>
wsil:httpHeader value="localhost:9080" key="Host" />
wsil:contentTypeHeader type="text" subtype="xml">
wsil:parameter key="charset" value="utf-8" quoted="true" />
 </wsil:contentTypeHeader>
 </wsil:httpHeaders>
wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0" containsProcessingInstructions="false" containsDTD="false">
s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:a="http://www.w3.org/2005/08/addressing">
s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
s:Fault xmlns:m="http://www.example.org/timeouts">
s:Code>
s:Value>s:Sender</s:Value>
s:Subcode>
s:Value>m:MessageTimeout</s:Value>
 </s:Subcode>
 </s:Code>
```

```
s:Reason>
s:Text xml:lang="en">Sender Timeout</s:Text>
 </s:Reason>
s:Detail>
m:MaxTime>P5M</m:MaxTime>
 </s:Detail>
 </s:Fault>
 </s:Body>
 </s:Envelope>
 </wsil:messageContents>
 </wsil:message>
 </wsil:messageLog>
wsil:descriptionFiles>
wsil:descriptionFile filename="../tns/corbaTypes.xsd">
   - <!--

                                schema for IDL to WSDL CORBA Namespace

                                Name: corbaTypes.xsd

 -->
xsd:schema xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
 xmlns:wsoap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:corba="http://www.omg.org/IDL-WSDL/1.0/"
 targetNamespace="http://www.omg.org/IDL-WSDL/1.0/">
xsd:element name="SourceIDL">
xsd:annotation>
xsd:documentation>IDL/WSDL Mapping Info</xsd:documentation>
 </xsd:annotation>
xsd:complexType>
xsd:sequence>
xsd:element name="source" type="xsd:string" minOccurs="1" maxOccurs="1" />
xsd:element name="version" type="xsd:string" minOccurs="1" maxOccurs="1" />
 </xsd:sequence>
 </xsd:complexType>
 </xsd:element>
```

```
xsd:element name="SourceRepositoryID">
xsd:annotation>
xsd:documentation>IDL Mapped Repository ID</xsd:documentation>
 </xsd:annotation>
xsd:complexType>
xsd:sequence>
xsd:element name="repositoryID" type="xsd:string" minOccurs="1" maxOccurs="1" />
xsd:element name="version" type="xsd:string" minOccurs="1" maxOccurs="1" />
 </xsd:sequence>
 </xsd:complexType>
 </xsd:element>
xsd:complexType name="ObjectReference">
xsd:sequence>
xsd:element name="url" type="xsd:anyURI" minOccurs="1" maxOccurs="unbounded" />
 </xsd:sequence>
 </xsd:complexType>
xsd:complexType name="CORBA.TypeCode">
xsd:sequence>
xsd:element name="definition" type="xsd:anyURI" maxOccurs="1" minOccurs="1" />
xsd:element name="typename" type="xsd:string" maxOccurs="1" minOccurs="1" />
 </xsd:sequence>
 </xsd:complexType>
xsd:complexType name="CORBA.Any">
xsd:sequence>
xsd:element name="type" type="corba:CORBA.TypeCode" maxOccurs="1" minOccurs="1" />
xsd:element name="value" type="xsd:anyType" maxOccurs="1" minOccurs="1" />
 </xsd:sequence>
 </xsd:complexType>
xsd:simpleType name="CORBA.completion_status">
xsd:restriction base="xsd:string">
xsd:enumeration value="COMPLETED_YES" />
xsd:enumeration value="COMPLETED_NO" />
xsd:enumeration value="COMPLETED_MAYBE" />
 </xsd:restriction>
 </xsd:simpleType>
```

```
xsd:complexType name="CORBA.SystemException">
xsd:sequence>
xsd:element name="minor" type="xsd:unsignedInt" maxOccurs="1" minOccurs="1" />
xsd:element name="completion_status" type="corba:CORBA.completion_status" maxOccurs="1" minOccurs="1" />
 </xsd:sequence>
 </xsd:complexType>
xsd:complexType name="_VALREF">
xsd:attribute name="ref" type="xsd:IDREF" use="optional" />
 <!--
empty attribute used for null semantics, i.e., value graph end nodes
 -->
 </xsd:complexType>
 </xsd:schema>
 </wsil:descriptionFile>
wsil:descriptionFile filename="../corba/corba.wsdl">
   - <!--

                                WSDL for IDL to WSDL CORBA Namespace

                                Name: corba.wsdl


 -->
definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:corba="http://www.omg.org/IDL-WSDL/1.0/" targetNamespace="http://www.omg.org/IDL-WSDL/1.0/" name="corba">
types>
xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
xsd:import namespace="http://www.omg.org/IDL-WSDL/1.0/" schemaLocation="../tns/corbaTypes.xsd" />
 </xsd:schema>
 </types>
message name="CORBA.SystemExceptionMessage">
part name="_return" type="corba:CORBA.SystemException" />
 </message>
 </definitions>
 </wsil:descriptionFile>
wsil:descriptionFile filename="../tns/SomeInterface2Example.wsdl">
   - <!--
```

```
                                WSDL Example

                                Name SomeInterface2Example.wsdl

                                mapping of the idl

                                /// IDL
                                typedef sequence<long> longSeq;
                                interface SomeInterface2 {
                                longSeq bar(in float pi);
                                };


-->
w:definitions xmlns:w="http://schemas.xmlsoap.org/wsdl/" xmlns:wsoap12="http://schemas.xmlsoap.org/wsdl/soap12/"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:corba="http://www.omg.org/IDL-WSDL/1.0/"
 xmlns:tns="http://www.omg.org/IDL-Mapped/" targetNamespace="http://www.omg.org/IDL-Mapped/"
 name="SomeInterface2Example">
w:import namespace="http://www.omg.org/IDL-WSDL/1.0/" location="../corba/corba.wsdl" />
w:types>
xsd:schema targetNamespace="http://www.omg.org/IDL-Mapped/">
 <!--
schema for soap encoding
 -->
xsd:complexType name="_SE_longSeq">
xsd:complexContent>
xsd:restriction base="soapenc:Array">
xsd:sequence>
xsd:element name="item" minOccurs="0" maxOccurs="unbounded" type="xsd:int" />
 </xsd:sequence>
   - <!--
    the following line, using the arrayType attribute from the wsdl namespace,
                                                                is specified to be used by WSDL1.1 for
   soapEncoding Array.  It is a strange incantation,
                                                                and the need to namespace qualify this
   attribute is why this file does not use wsdl as the default namespace
```

```
-->
xsd:attribute ref="soapenc:arrayType" w:arrayType="xsd:int" />
</xsd:restriction>
</xsd:complexContent>
<!--
schema for WS-I compliant mapping
-->
</xsd:complexType>
xsd:complexType name="longSeq">
xsd:sequence>
xsd:element name="item" minOccurs="0" maxOccurs="unbounded" type="xsd:int" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
</w:types>
w:message name="SomeInterface2.bar">
w:part name="pi" type="xsd:float" />
</w:message>
w:message name="SomeInterface2.barResponse">
w:part name="_return" type="tns:longSeq" />
</w:message>
w:message name="_SE_SomeInterface2.barResponse">
w:part name="_return" type="tns:_SE_longSeq" />
</w:message>
w:portType name="_SE_SomeInterface2">
w:operation name="bar">
w:input message="tns:SomeInterface2.bar" />
w:output message="tns:_SE_SomeInterface2.barResponse" />
w:fault name="CORBA.SystemException" message="corba:CORBA.SystemExceptionMessage" />
</w:operation>
</w:portType>
w:portType name="SomeInterface2">
w:operation name="bar">
w:input message="tns:SomeInterface2.bar" />
w:output message="tns:SomeInterface2.barResponse" />
```

```
w:fault name="CORBA.SystemException" message="corba:CORBA.SystemExceptionMessage" />
 </w:operation>
 </w:portType>
 </w:definitions>
 </wsil:descriptionFile>
wsil:descriptionFile filename="SomeInterface2BindingExample.wsdl">
   - <!--

                                WSDL Example

                                Name SomeInterface2BindingExample.wsdl

                                soap encoding mapping binding of the idl

                                /// IDL
                                typedef sequence<long> longSeq;
                                interface SomeInterface2 {
                                longSeq bar(in float pi);
                                };


 -->
definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:wsoap12="http://schemas.xmlsoap.org/wsdl/soap12/"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:corba="http://www.omg.org/IDL-WSDL/1.0/"
 xmlns:tns="http://www.omg.org/IDL-Mapped/" targetNamespace="http://www.omg.org/IDL-Mapped/"
 name="SomeInterface2BindingExample">
import namespace="http://www.omg.org/IDL-Mapped/" location="../tns/SomeInterface2Example.wsdl" />
binding name="_SE_SomeInterface2Binding" type="tns:_SE_SomeInterface2">
wsoap12:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
operation name="bar">
wsoap12:operation soapAction="SomeInterface2#bar" />
input>
wsoap12:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://www.omg.org/ID
 WSDL/1.0/" />
 </input>
output>
```

```
wsoap12:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://www.omg.org/ID
 WSDL/1.0/" />
 </output>
fault name="CORBA.SystemException">
wsoap12:fault name="CORBA.SystemException" use="literal" />
 </fault>
 </operation>
 </binding>
binding name="SomeInterface2Binding" type="tns:SomeInterface2">
wsoap12:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
operation name="bar">
wsoap12:operation soapAction="SomeInterface2#foo" />
input>
wsoap12:body use="literal" namespace="http://www.omg.org/IDL-WSDL/1.0/" />
 </input>
output>
wsoap12:body use="literal" namespace="http://www.omg.org/IDL-WSDL/1.0/" />
 </output>
fault name="CORBA.SystemException">
wsoap12:fault name="CORBA.SystemException" use="literal" />
 </fault>
 </operation>
 </binding>
 </definitions>
 </wsil:descriptionFile>
wsil:descriptionFile filename="aiag.xsd">
xsd:schema xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
 xmlns:wsoap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.openapplications.org/oagis/9/aiag/2">
xsd:element name="SyncReceiveDelivery" type="xsd:string" />
xsd:element name="SyncQuantityOnHand2" type="xsd:string" />
 </xsd:schema>
 </wsil:descriptionFile>
wsil:descriptionFile filename="nons.xsd">
xsd:schema xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
 xmlns:wsoap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
```

```
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
xsd:element name="SynchA" type="xsd:string" />
 </xsd:schema>
 </wsil:descriptionFile>
wsil:descriptionFile filename="ISupplier.wsdl">
wsdl:definitions xmlns:sup="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:aiag2="http://www.openapplications.org/oagis/9/aiag/2"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="ICustomer"
 targetNamespace="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0"
 xmlns:tns="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0"
 xmlns:aiagimp="http://www.openapplications.org/oagis/9/aiag/2"
 xmlns:wsoap12="http://schemas.xmlsoap.org/wsdl/soap12/">
wsdl:types>
xsd:schema>
xsd:import namespace="http://www.openapplications.org/oagis/9/aiag/2" schemaLocation="aiag.xsd" />
xsd:import namespace="" schemaLocation="nons.xsd" />
 </xsd:schema>
 </wsdl:types>
wsdl:message name="SyncQuantityOnHandMessage">
wsdl:part element="SynchA" name="OHpart1" />
wsdl:part element="aiagimp:SyncQuantityOnHand2" name="OHpart2" />
 </wsdl:message>
wsdl:message name="SyncReceiveDeliveryMessage">
wsdl:part element="aiagimp:SyncReceiveDelivery" name="SRDpart" />
 </wsdl:message>
wsdl:message name="SyncQuantityOnFootMessage">
wsdl:part element="SyncQuantityOnFootHeader" name="SQOFTheader" />
wsdl:part element="SyncQuantityOnFootHeader" name="SQFLTheader" />
wsdl:part element="tns:SyncQuantityOnFoot" name="SQOFpart" />
 </wsdl:message>
wsdl:message name="SyncQuantityOnHeadMessage">
wsdl:part element="xsd:string" name="SQOH1" />
wsdl:part element="xsd:string" name="SQOH2" />
wsdl:part element="xsd:string" name="SQOH3" />
wsdl:part element="xsd:string" name="HeadSQOH" />
wsdl:part element="xsd:string" name="HFSQOH" />
```

```
</wsdl:message>
wsdl:message name="SyncQuantityOnHeadResponse">
wsdl:part element="xsd:string" name="SQOH1" />
wsdl:part element="xsd:string" name="SQOH4" />
wsdl:part element="xsd:string" name="SQOH5" />
</wsdl:message>
wsdl:message name="SyncQuantityOnHeadFault">
wsdl:part element="xsd:string" name="SQOHF1" />
<!--
wsdl:part element="xsd:string" name="SQOHF2"/
-->
</wsdl:message>
wsdl:message name="SyncQuantityOnHeadFault2">
wsdl:part element="xsd:string" name="SQOHF2b" />
</wsdl:message>
wsdl:portType name="sup:ISupplier">
wsdl:operation name="SyncReceiveDelivery">
wsdl:input message="tns:SyncReceiveDeliveryMessage" />
</wsdl:operation>
wsdl:operation name="SyncQuantityOnHand" parameterOrder="SyncQuantityOnHand">
wsdl:input message="SyncQuantityOnHandMessage" />
</wsdl:operation>
wsdl:operation name="SyncQuantityOnFoot">
wsdl:input message="SyncQuantityOnFootMessage" />
</wsdl:operation>
wsdl:operation name="SyncQuantityOnHead" parameterOrder="SQOH1 SQOH3">
wsdl:input message="SyncQuantityOnHeadMessage" />
wsdl:output message="SyncQuantityOnHeadResponse" />
wsdl:fault name="Fault1" message="SyncQuantityOnHeadFault" />
wsdl:fault name="Fault2" message="SyncQuantityOnHeadFault2" />
</wsdl:operation>
</wsdl:portType>
wsdl:binding name="ISupplierSOAP" type="sup:ISupplier">
wsoap12:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
<!--
wsoap12:binding style="document" transport="http://schemas.xmlsoap.org/soap/https" /
```

```
-->
wsdl:operation name="SyncReceiveDelivery">
wsoap12:operation soapAction="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncReceiveDelivery" /
wsdl:input>
wsoap12:body use="literal" />
<!--
wsoap12:body parts="ApplicationArea DataArea" use="literal" /
-->
</wsdl:input>
</wsdl:operation>
wsdl:operation name="SyncQuantityOnHand">
wsoap12:operation soapAction="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncQuantityOnHand" /
wsdl:input>
wsoap12:body parts="OHpart1 OHpart2" use="literal" />
</wsdl:input>
</wsdl:operation>
wsdl:operation name="SyncQuantityOnFoot">
wsoap12:operation soapAction="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncQuantityOnFoot" /
wsdl:input>
wsoap12:header message="SyncQuantityOnFootMessage" part="SQOFTheader" use="literal" />
wsoap12:headerfault message="SyncQuantityOnFootMessage" part="SQFLTheader" use="literal" />
wsoap12:body parts="SQOFpart" use="literal" />
</wsdl:input>
  - <!--
   wsdl:fault>
                                                    <wsoap12:fault parts="SQOFheader" use="literal" />
                                                    </wsdl:fault

-->
</wsdl:operation>
wsdl:operation name="SyncQuantityOnHead">
wsoap12:operation soapAction="http://www.openapplications.org/aiag2/interface/minmax/supplier/v1.0/SyncQuantityOnHead" /
wsdl:input>
wsoap12:header message="SyncQuantityOnHeadMessage" part="HeadSQOH" use="literal" />
wsoap12:body parts="SQOH1 SQOH2 SQOH3" use="literal" />
</wsdl:input>
wsdl:output>
```

```
wsoap12:header message="SyncQuantityOnHeadMessage" part="HeadSQOH" use="literal" />
wsoap12:body parts="SQOH1 SQOH4 SQOH5" use="literal" />
wsoap12:headerfault message="SyncQuantityOnHeadMessage" part="HFSQOH" use="literal" />
 </wsdl:output>
wsdl:fault>
wsoap12:fault name="Fault1" use="literal" />
 </wsdl:fault>
wsdl:fault>
wsoap12:fault name="Fault2" use="literal" />
 </wsdl:fault>
 </wsdl:operation>
 </wsdl:binding>
wsdl:service name="SupplierService">
wsdl:port name="ISupplier" binding="sup:ISupplierSOAP">
wsoap12:address location="http://www.example.org/" />
 </wsdl:port>
 </wsdl:service>
 </wsdl:definitions>
 </wsil:descriptionFile>
 </wsil:descriptionFiles>
 </wsil:testLog>
```

Figure 3 – Example of Transformed Message Log file.

6.3 Appendix C: Q & A on What to Expect from the Testing Tools

Question: Can the testing tools guarantee that a Web Service is conforming to the Profile or an applicable conformance scope?
Answer: The tools can only verify the conformance of Web Service *artifacts* that are produced during a testing session. Some artifacts belong to the definition of the Web Service (WSDL); some others result from the observable behavior of the Web Service at run-time. It is difficult to test all possible behaviors that a Web Service can exhibit, mostly because exercising these behaviors is application-dependent and requires an application-level understanding of the Web Service. The testing tools are then an *indicator* of conformance of a Web Service to the Basic Profile or an applicable conformance scope, based on the artifacts produced. In turn, this is an indicator of interoperability with other business partners who also have tested as conformant.

Question: Can the testing tools verify all the requirements of the Basic Profile or an applicable conformance scope?
Answer: No. A few requirements of the WS-I basic profile cannot be easily tested, and have no corresponding test assertion. Such requirements fall into one of these categories:

- The profile requirement refers to an external specification document that is too complex to test, for an outcome that has been prioritized as low, given current resources. An example is the requirement on cookies which, when used, must conform to RFC2965.
- The requirement is not possible to test using the current test harness. For example, requirements about the HTTP code value when a request has been redirected.
- The requirement is about interpretative behavior of a Web Service consumer or provider, which exceeds the capability of the test harness, and would require more intrusive technology, or more knowledge of the WS application and semantics.

This is another reason why the tools should be defined as an indicator of conformance, rather than as certification tools. However the tools provide a powerful indicator of the ability of a Web Service to interoperate with any external party known to also comply with the Profile.

Question: How can we be sure that all the operations of a Web Service have been covered in the testing?
Answer: This depends on the Test Scenarios used for this service. A complete coverage of all the Web Service operations will rely on the client program involved in the testing of the WS, which is either ad-hoc, or is a real application in deployment over which the test operator does not have much control. Even so, such a driver may not be able to trigger an exhaustive set of behaviors, e.g. those inducing all kinds of faults.

Question: What are some practical situations where the testing tools show value to Web Services users or vendors?

Answer: An industry may define industry-specific Web Services – e.g. purchase order submission, request for product information - and specific usage scenarios. This industry may require that the Web Service, when used according to these expected interaction scenarios, exhibits a profile-conforming behavior, as verified by WS-I testing tools. In order to achieve this, this industry will likely define a specific test driver for its Web Services. By doing so, this industry has effectively defined an industry-specific test harness and certification criterion for interoperability, based on the Profile. If such a Web Service passes the tests, a vendor in this industry can claim that it is interoperable with any user application, provided that the user also complies with the Profile, and exercises the expected usage scenarios.

Another scenario shows value for interoperability trouble shooting: a client application may fail to interoperate with a Web Service, although both claim to be conforming to the Basic Profile. Because the testing tools can monitor messages from both interacting parties, the tools can be used to diagnose a failure to interoperate, and to identify the cause: either the client application or the Web Service may exhibit non-conforming behavior during this particular interaction. This will help determine responsibilities.

Question: Will the WS-I Test Framework also support functional testing of a particular Web Service?
Answer: This is outside of the scope of conformance testing to part or all of the Basic Profile. Such testing would involve knowledge of the application semantics that is specific to each Web Service. The Monitor developed is reusable to provide the message capture necessary to such testing.

Acknowledgements

Robert Chumbley, IBM; Test methods and tools
Jacques Durand and Tom Rutt, Fujitsu: Test analyzer and assertion development

