Basic Profile 2.0 Test Scenarios

Working Group Draft

2008-11-24

This version:
Same
Latest version:
Same
Last version:
http://tinyurl.com/5zhaj6 (11/18/2008)

Editors:
Robert Chumbley, IBM
Monica J. Martin, Microsoft

Administrative contact:
secretary@ws-i.org

1.1 Abstract

This document provides a set of scenarios for testing interoperability under Basic Profile 2.0 focused on SOAP, WSDL, MTOM and WS-Addressing. Tests for simple and complex data types are also included that leverage different SOAP binding styles and uses (rpc-literal and doc-literal) in SOAP 1.2. The WS-Addressing Test Suite is used in additional to other scenarios to effectively test Basic Profile 2. Status of this Document

This document is an Editor's draft in this Working Group. It is a work in progress, and should not be considered as final; other documents will supersede this document.

1.2 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.

1.3 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 wsbasic_comment@ws-i.org.

Table of Contents

2 Introduction

This document provides a set of scenarios for testing interoperability under Basic Profile 2.0. It is intended to provide the information necessary to implement the described scenarios. The majority of scenarios come from well-established interoperation test suites for the underlying specifications that make up Basic Profile 2.0. These scenarios are as follows:

- Schema and WSDL Data Type Tests
  - Base data types, document-literal and rpc-literal encoding
  - Complex data types, document-literal and rpc-literal encoding
- MTOM Tests
  - MTOM Scenarios, SOAP 1.2 endpoint
- Web Services Addressing 1.0 Test Suite (http://www.w3.org/2002/ws/addr/testsuite/)
  - All Required SOAP 1.2 tests (test12xx).
- Additional tests to effectively test Basic Profile 2.0 requirements.

This document describes these scenarios in detail, including a description of SENDER actions, RECEIVER operations, and sample message exchanges.

3 Schema and WSDL Tests

Interoperability scenarios for various data types, SOAP binding style and binding use are provided. The scenarios are described in terms of XML Schema. Examples messages are provided with these scenarios.

The scenarios are further categorized based on the data type. The basic data type scenarios tests the simple XML schema data type such as integer, long, string etc. The complex data types include arrays. For the two data types, the supported binding styles and use (rpc-literal and doc-literal) will be tested.

Example messages and WSDL are provided with these scenarios.

3.1 Test Procedure

All scenarios have two participants, SENDER and RECEIVER, consistent with the roles defined in Basic Profile 2.0 (See Section 2.2 Conformance Targets). All of the scenarios are “echo” type. The SENDER sends the test value to the RECEIVER and the RECEIVER echoes the same value back.

The SENDER starts the test by invoking the appropriate operation on the RECEIVER with a scenario test value. The RECEIVER replies with the same value back to the SENDER. The SENDER compares the value as per the data type being tested. If the value received by the SENDER is same as the one sent, the scenario is considered a successful test for that particular test value. Each scenario has more than one test value and for the entire scenario to be successful, all test values must be correct.

3.2 Flow of Testing Schema and WSDL Scenarios

The flow of testing these scenarios is:

1. Generate the SENDER consuming the RECEIVER WSDL
2. Implement the SENDER and for each scenario send the appropriate test value
3. Compare the return value from the RECEIVER and declare the scenario as successful.

3.3 Base Data Types

The base data type scenarios test simple data types. There are 2 services each providing the scenarios for the same data types with different binding style and use.

1. Document / Literal / Wrapped
2. RPC / Literal

Scenario Name	**XSD Type**	**Test Values and Example Messages**	
		Doc / Lit / Wrapped	RPC / Lit
RetBoolean	xs:Boolean	Click here	Click here
RetUnsignedByte	xs:unsignedByte	Click here	Click here
RetByte	xs:byte	Click here	Click here
RetBase64Binary	xs:base64Binary	Click here	Click here
RetDecimal	xs:decimal	Click here	Click here
RetFloat	xs:float	Click here	Click here
RetDouble	xs:double	Click here	Click here
RetInt	xs:int	Click here	Click here
RetShort	xs:short	Click here	Click here
RetLong	xs:long	Click here	Click here
RetAnyType* (see explanation below)	xs:anyType	Click here	Click here
RetUnsignedInt	xs:unsignedInt	Click here	Click here
RetUnsignedShort	xs:unsignedShort	Click here	Click here

RetUnsignedLong	xs:unsignedLong	Click here	Click here
RetString	xs:string	Click here	Click here
RetAnyUri	xs:anyURI	Click here	Click here
RetDateTime	xs:dateTime	Click here	Click here
RetDuration	xs:duration	Click here	Click here
RetQName	xs:QName	Click here	Click here

3.3.1 RetAnyType Test Description

The RetAnyType test simulates user scenarios in which a request or response may include a value for a type taken from a set of known types. In each message exchange, a value representing a basic XML Schema type is transmitted on the wire using xs:anyType as the base type, with the xsi:type attribute (http://www.w3.org/2001/XMLSchema-instance) specifying the actual type that should be used to interpret the value in the message. The value is reflected by the service, using the same xsi:type information. Values transmitted in this test should be taken from the following set of XML Schema types: base64Binary, boolean, byte, decimal, float, double, int, long, QName, short, and string.

3.4 Complex Data Types

The complex data type scenarios test the complex data types (arrays). There are 2 services each providing the scenarios for the same data types with different binding style and use.

Scenario Name	**Test Values and Example Messages**	
	Doc / Lit / Wrapped	RPC / Lit
RetArrayString1D	Click here	Click here
RetArrayInt1D	Click here	Click here

3.5 Sample Messages for Schema and WSDL Tests

In addition to the data types and SENDER and RECEIVER actions for these tests described in the previous sections, sample messages for the new services are provided in the attached messages directory. The following HTML pages index the messages:

Test 2800 BaseDataTypes.DocLitWrapped.htm [Sample WSDL is included in the accompanying zip file at WSDL\SchemaAndWSDL\Soap12\BaseDataTypesDocLitW.wsdl]
Test 2810 BaseDataTypes.RpcLit.htm [Sample WSDL is included in the accompanying zip file at WSDL\SchemaAndWSDL\Soap12\BaseDataTypesRpcLit.wsdl]

Test 2830 ComplexDataTypes.DocLitW.htm [Sample WSDL is included in the accompanying zip file at WSDL\SchemaAndWSDL\Soap12\ComplexDataTypesDocLitW.wsdl]
Test 2840 ComplexDataTypes.RpcLit.htm [Sample WSDL is included in the accompanying zip file at WSDL\SchemaAndWSDL\Soap12\ComplexDataTypesRpcLit.wsdl]

3.6 Test 2801 (RetString UTF-16)

The description for this scenario is encoded using UTF-16 text encoding. This test uses a document-literal binding with SOAP 1.2 and the same portType described for the RetString operation above. The SENDER sends a series of messages containing string in the body to the RECEIVER. The RECEIVER echoes the strings back in the HTTP response.

3.6.1 Sample SENDER Request Message

```
<wsil:testLog xmlns:wsil="http://www.ws-i.org/testing/2008/02/log/">
<wsil:messageLog>
<wsil:message type="request" id="1" conversation="1">
<wsil:httpHeaders>
  <wsil:requestLine>POST /BaseDataTypes HTTP/1.1</wsil:requestLine>
  <wsil:contentTypeHeader type="application" subtype="soap+xml">
    <wsil:parameter value="utf-16" key="charset"/>
  </wsil:contentTypeHeader>
</wsil:httpHeaders>
<wsil:messageContents encoding="utf-16" validXml="true" xmlVersion="1.0"
containsProcessingInstructions="false" containsDTD="false">
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IBaseDataTypesDocLitW/RetString</a:Ac
tion>
    <a:MessageID>urn:uuid:f8a4a5fa-a110-437a-a391-9f76a95deab4</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://example.org/BaseDataTypes </a:To>
  </s:Header>
  <s:Body>
    <RetString xmlns="http://tempuri.org/">
      <inString>Hello World</inString>
    </RetString>
  </s:Body>
</s:Envelope>
</wsil:messageContents>
</wsil:message>
</wsil:messageLog>
</wsil:testLog>
```

3.6.2 Sample RECEIVER Response Message

```
<wsil:testLog xmlns:wsil="http://www.ws-i.org/testing/2008/02/log/">
<wsil:messageLog>
<wsil:message type="response" id="1" conversation="1">
<wsil:httpHeaders>
  <wsil:requestLine>HTTP/1.1 200 OK</wsil:requestLine>
```

```
  <wsil:contentTypeHeader type="application" subtype="soap+xml">
    <wsil:parameter value="utf-16" key="charset"/>
  </wsil:contentTypeHeader>
</wsil:httpHeaders>
<wsil:messageContents encoding="utf-16" validXml="true" xmlVersion="1.0"
containsProcessingInstructions="false" containsDTD="false">
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IBaseDataTypesDocLitW/RetStringRespon
se</a:Action>
    <a:RelatesTo>urn:uuid:f8a4a5fa-a110-437a-a391-9f76a95deab4</a:RelatesTo>
  </s:Header>
  <s:Body>
    <RetStringResponse xmlns="http://tempuri.org/">
      <RetStringResult>Hello World</RetStringResult>
    </RetStringResponse>
  </s:Body>
</s:Envelope>
</wsil:messageContents>
</wsil:message>
</wsil:messageLog>
</wsil:testLog>
```

4 MTOM Tests

The tests for MTOM messaging are described in the sections that follow. For BP 2.0 compliance testing, SOAP 1.2, WS-Addressing 1.0 and UTF-8 encoding are used.

All scenarios have two participants, SENDER and RECEIVER. All scenarios use the same Service Contract.

These MTOM test scenarios cover essential combinations of MTOM encoding applied to different data structures and character encodings. The first four tests cover optimizing binary data in various parts of a message.

Message Examples for these scenarios are available in the attached “Messages” directory (Mtom-wire.txt).

Sample WSDL for these tests is provided in the accompanying zip file (WSDL\MTOM\Soap12MtomUtf8.wsdl)

4.1 Test2902 Echo Binary As String

Request has a XOPed binary that contains UTF-8 encoded text, response contains the passed string.

4.2 Test2904 Echo String As Binary

Request has a string that is returned back as XOPed binary blob (UTF-8 encoded).

4.3 Test 2906 Echo Array of Binaries As Array Of Strings

Request contains array, each array element has type base64Binary, XOPed, contains unique UTF-8 encoded strings. Response has array of strings matching those in the request.

4.4 Test2908 Echo Binary Field As String

Request contains a structure, one of the fields is binary, contains a string UTF-8 encoded. Response contains a string from the binary field from the request. See:

<Body>
<EchoBinaryFieldAsString…>
<Name> xs:string </Name>
<Array>…</Array>
</EchoBinaryFieldAsString>
</Body>

4.5 Test 2900 (EchoStringArrayAsBinaryArray UTF-8)

Ensures that base64Binary array values are correctly optimized by the RECEIVER and correctly decoded by the SENDER. The RECEIVER endpoint uses a SOAP 1.2 binding with WS-Addressing 1.0 and UTF-8 encoding. The SENDER sends an array of strings. The RECEIVER encodes this string array as an array of base64Binary values using UTF-8 encoding and returns the result on the HTTP response. The test passes if the data in the arrays match after decoding.

4.5.1 Sample Message Exchange

The messages represent the logical SOAP messages before optimization is applied.

4.5.1.1 Sample SENDER Request

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://xmlsoap.org/EchoStringArrayAsBinaryAr
ray</a:Action>
    <a:MessageID>urn:uuid:6417bac4-f352-4bcd-b25b-
2d39c9541dc2</a:MessageID>
    <a:ReplyTo>
<a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To
s:mustUnderstand="1">http://www.example.org/Soap12MtomUtf8</a:To>
  </s:Header>
  <s:Body>
    <EchoStringArrayAsBinaryArray
xmlns="http://xmlsoap.org/Ping">
      <stringArray xmlns:d4p1="http://schemas.example.org/Arrays"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <d4p1:string><string-data></d4p1:string>
        …
        <d4p1:string><string-data></d4p1:string>
      </stringArray>
```

```
    </EchoStringArrayAsBinaryArray>
  </s:Body>
</s:Envelope>
```

4.5.1.2 Sample RECEIVER Response

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://xmlsoap.org/EchoStringArrayAsBinaryAr
rayReply</a:Action>
    <a:RelatesTo>urn:uuid:6417bac4-f352-4bcd-b25b-
2d39c9541dc2</a:RelatesTo>
  </s:Header>
  <s:Body>
    <EchoStringArrayAsBinaryArrayResponse
xmlns="http://xmlsoap.org/Ping">
      <EchoStringArrayAsBinaryArrayResult
xmlns:b="http://schemas.example.org/Arrays"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <b:base64Binary><base64binary-data></b:base64Binary>
        …
        <b:base64Binary><base64binary-data></b:base64Binary>
      </EchoStringArrayAsBinaryArrayResult>
    </EchoStringArrayAsBinaryArrayResponse>
  </s:Body>
</s:Envelope>
```

4.6 Test 2910 (EchoStringArrayAsBinaryFields UTF-8)

Ensures that base64Binary values embedded in structures are correctly optimized by the RECEIVER and correctly decoded by the SENDER. The RECEIVER endpoint uses a SOAP 1.2 binding with WS-Addressing 1.0 and UTF-8 encoding. The SENDER sends an array of four strings. The RECEIVER encodes this string array in a structure with four base64Binary fields using UTF-8 encoding and returns the result on the HTTP response. The test passes if the data in the structure matches the data in the array. The structure sent by the RECEIVER is of the following form:

```
<EchoStringArrayAsBinaryFieldsResult >
  <FirstBinaryValue><base64binary-data></FirstBinaryValue>
  <FourthBinaryValue><base64binary-data></FourthBinaryValue>
  <Name><string-data></Name>
  <SecondBinaryValue><base64binary-data></SecondBinaryValue>
  <ThirdBinaryValue><base64binary-data></ThirdBinaryValue>
</EchoStringArrayAsBinaryFieldsResult>
```

4.6.1 Sample Message Exchange

The messages below represent the logical SOAP messages before optimization is applied:

4.6.1.1 Sample SENDER Request

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
```

```
    <a:Action
s:mustUnderstand="1">http://xmlsoap.org/EchoStringArrayAsBinaryFi
elds</a:Action>
    <a:MessageID>urn:uuid:d413cdfc-9995-44bc-874c-
48d7634082c0</a:MessageID>
    <a:ReplyTo>
<a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addr
ess>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/
Soap12MtomUtf8</a:To>
  </s:Header>
  <s:Body>
    <EchoStringArrayAsBinaryFields
xmlns="http://xmlsoap.org/Ping">
      <textArray xmlns:d4p1="http://schemas.example.org/Arrays"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <d4p1:string><string-data></d4p1:string>
        <d4p1:string><string-data></d4p1:string>
        <d4p1:string><string-data></d4p1:string>
        <d4p1:string><string-data></d4p1:string>
      </textArray>
    </EchoStringArrayAsBinaryFields>
  </s:Body>
</s:Envelope>
```

4.6.1.2 Sample RECEIVER Response

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://xmlsoap.org/EchoStringArrayAsBinaryFi
eldsReply</a:Action>
    <a:RelatesTo>urn:uuid:d413cdfc-9995-44bc-874c-
48d7634082c0</a:RelatesTo>
  </s:Header>
  <s:Body>
    <EchoStringArrayAsBinaryFieldsResponse
xmlns="http://xmlsoap.org/Ping">
      <EchoStringArrayAsBinaryFieldsResult
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <FirstBinaryValue><base64binary-data></FirstBinaryValue>
        <FourthBinaryValue><base64binary-
data></FourthBinaryValue>
        <Name>ReturnStructure</Name>
        <SecondBinaryValue><base64binary-
data></SecondBinaryValue>
        <ThirdBinaryValue><base64binary-data></ThirdBinaryValue>
      </EchoStringArrayAsBinaryFieldsResult>
    </EchoStringArrayAsBinaryFieldsResponse>
  </s:Body>
</s:Envelope>
```

4.7 Test 2930 Echo Binary Header

Ensures that base64Binary values embedded in SOAP headers are correctly optimized and decoded by both SENDER and RECEIVER. The RECEIVER uses a SOAP 1.2 binding with WS-Addressing 1.0 and UTF-8 encoding. The SENDER creates two text values, encodes one value using UTF-8 encoding and places the value in the header of the SOAP request, as the content of a namespace-qualified XML element. The SENDER places the other text value in the body of the SOAP. The RECEIVER retrieves both values from the SOAP request, decodes the base64Binary value from the header and places the resulting text value in the body of the response, encodes the text value from the request body using UTF-8 encoding, and places the resulting base64Binary value in the header of the response as the content of a namespace-qualified XML element. The accompanying service description provides schema for the request and response messages.

4.7.1 Sample Message Exchange

The messages below represent the logical SOAP messages before optimization is applied.

4.7.1.1 Sample SENDER Request

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://xmlsoap.org/HeaderRequest</a:Action>
    <h:HeaderData xmlns:h="http://xmlsoap.org/Ping">
      <base64binary-data A>
    </h:HeaderData>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:MessageID>urn:uuid:0818c1f5-4c1e-4a78-8d8f-1fdcbc18f2a3</a:MessageID>
    <a:To s:mustUnderstand="1">http://www.example.org/Soap12MtomUtf8</a:To>
  </s:Header>
  <s:Body>
    <EchoBinaryHeaderRequest xmlns="http://xmlsoap.org/Ping">
      <BodyData><string-data B></BodyData>
    </EchoBinaryHeaderRequest>
  </s:Body>
</s:Envelope>
```

4.7.1.2 Sample RECEIVER Response

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://xmlsoap.org/HeaderResponse</a:Action>
    <h:HeaderData xmlns:h="http://xmlsoap.org/Ping">
      <base64binary-data B>
    </h:HeaderData>
    <a:RelatesTo>urn:uuid:0818c1f5-4c1e-4a78-8d8f-1fdcbc18f2a3</a:RelatesTo>
  </s:Header>
  <s:Body>
    <EchoBinaryHeaderResponse xmlns="http://xmlsoap.org/Ping">
      <BodyData><string-data A></BodyData>
```

```
    </EchoBinaryHeaderResponse>
  </s:Body>
</s:Envelope>
```

4.8 Test 2901 (UTF-16 Test)

Test 2900 is repeated using a SENDER and RECEIVER that support UTF-16 encoding. The published Description for the RECEIVER is encoded using UTF-16 encoding, as well as all exchanged messages. Message exchange is exactly as described above (using only a different To EPR).

5 New WS-Addressing Tests

5.1 Use of WS-Addressing and SOAP 1.2

The WS-Addressing 1.0 Test Suite is used as a base to refine a new scenario to test WS-Addressing 1.0 and SOAP 1.2.

5.1.1 Summary

A series of request-response exchanges are used to exercise WS-Addressing and SOAP 1.2. These tests are similar to WS-Addressing tests 1090, 1130 and 1133. A refined WS-Addressing Test Service WSDL is found in the .zip file for this scenario package.

5.1.2 Message Exchange

The message exchange is as follows:

- SENDER sends Echos using an anonymous ReplyTo. The RECEIVER mirrors the requested Echo in the response.
- SENDER sends Echo request using an anonymous ReplyTo.
- RECEIVER responds with Echo to that anonymous address.
- SENDER sends another Echo In to node B with anonymous ReplyTo.
- RECEIVER responds with Echo Out to that anonymous address.
- SENDER sends another Echo In again to node B with an anonymous ReplyTo.
- RECEIVER responds with Echo out.

5.1.2.1 Sample SENDER Request 1

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope">
    <S:Header>
        <To
xmlns="http://www.w3.org/2005/08/addressing">http://example.org/node/B</To>
        <Action xmlns="http://www.w3.org/2005/08/addressing">
            http://example.org/wsaTestService/wsaTestPortType/echo1Request
        </Action>
        <ReplyTo xmlns="http://www.w3.org/2005/08/addressing">
            <Address>http://www.w3.org/2005/08/addressing/anonymous</Address>
```

```
        </ReplyTo>
        <MessageID
xmlns="http://www.w3.org/2005/08/addressing">uuid:fc056668-8058-4e94-bdea-
3ca58a476533
        </MessageID>
    </S:Header>
    <S:Body>
        <echoIn1 xmlns="http://example.org/echo">Echo This Text</echoIn1>
    </S:Body>
</S:Envelope>
```

5.1.2.2 Sample RECEIVER Response 1

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope">
    <S:Header>
        <To
xmlns="http://www.w3.org/2005/08/addressing">http://www.w3.org/2005/08/addres
sing/anonymous</To>
        <Action xmlns="http://www.w3.org/2005/08/addressing">
            http://example.org/wsaTestService/wsaTestPortType/echo1Response
        </Action>
        <MessageID
xmlns="http://www.w3.org/2005/08/addressing">uuid:40c04b0f-ee1f-49a2-afad-
3d532946c1a9
        </MessageID>
        <RelatesTo
xmlns="http://www.w3.org/2005/08/addressing">uuid:fc056668-8058-4e94-bdea-
3ca58a476533
        </RelatesTo>
    </S:Header>
    <S:Body>
        <echoOut1 xmlns="http://example.org/echo">Echo This Text</echoOut1>
    </S:Body>
</S:Envelope>
```

5.1.2.3 Sample SENDER Request 2

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope">
    <S:Header>
        <To
xmlns="http://www.w3.org/2005/08/addressing">http://example.org/node/B</To>
        <Action xmlns="http://www.w3.org/2005/08/addressing">
            http://example.org/wsaTestService/wsaTestPortType/echo2In
        </Action>
        <ReplyTo xmlns="http://www.w3.org/2005/08/addressing">
            <Address>http://www.w3.org/2005/08/addressing/anonymous</Address>
        </ReplyTo>
        <MessageID
xmlns="http://www.w3.org/2005/08/addressing">uuid:e2855e5f-2759-4048-b4f1-
48d077cb1ff4
        </MessageID>
    </S:Header>
    <S:Body>
        <echoIn2 xmlns="http://example.org/echo">Echo This Text</echoIn2>
    </S:Body>
</S:Envelope>
```

5.1.2.4 Sample RECEIVER Response 2

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope">
    <S:Header>
        <To
xmlns="http://www.w3.org/2005/08/addressing">http://www.w3.org/2005/08/addres
sing/anonymous</To>
        <Action xmlns="http://www.w3.org/2005/08/addressing">
            http://example.org/wsaTestService/wsaTestPortType/echo2Out
        </Action>
        <MessageID
xmlns="http://www.w3.org/2005/08/addressing">uuid:8bd69f5d-b9b8-40e0-b32e-
796729ff4a31
        </MessageID>
        <RelatesTo
xmlns="http://www.w3.org/2005/08/addressing">uuid:e2855e5f-2759-4048-b4f1-
48d077cb1ff4
        </RelatesTo>
    </S:Header>
    <S:Body>
        <echoOut2 xmlns="http://example.org/echo">Echo This Text</echoOut2>
    </S:Body>
</S:Envelope>
```

5.1.2.5 Sample SENDER Request 3

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope">
    <S:Header>
        <To
xmlns="http://www.w3.org/2005/08/addressing">http://example.org/node/B</To>
        <Action
xmlns="http://www.w3.org/2005/08/addressing">http://example.org/action/echo3I
n</Action>
        <ReplyTo xmlns="http://www.w3.org/2005/08/addressing">
            <Address>http://www.w3.org/2005/08/addressing/anonymous</Address>
        </ReplyTo>
        <MessageID
xmlns="http://www.w3.org/2005/08/addressing">uuid:b1411672-fdbf-4482-9e92-
d76b41e8f0ed
        </MessageID>
    </S:Header>
    <S:Body>
        <echoIn3 xmlns="http://example.org/echo">Echo This Text</echoIn3>
    </S:Body>
</S:Envelope>
```

5.1.2.6 Sample RECEIVER Response 3

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope">
    <S:Header>
        <To
xmlns="http://www.w3.org/2005/08/addressing">http://www.w3.org/2005/08/addres
sing/anonymous</To>
        <Action
xmlns="http://www.w3.org/2005/08/addressing">http://example.org/action/echo3O
ut</Action>
        <MessageID
xmlns="http://www.w3.org/2005/08/addressing">uuid:e94bb483-aa06-45b5-85ba-
9435857f9206
```

```
        </MessageID>
        <RelatesTo
xmlns="http://www.w3.org/2005/08/addressing">uuid:b1411672-fdbf-4482-9e92-
d76b41e8f0ed
        </RelatesTo>
    </S:Header>
    <S:Body>
        <echoOut3 xmlns="http://example.org/echo">Echo This Text</echoOut3>
    </S:Body>
</S:Envelope>
```

5.2 Test 1291

Test for correct response when MessageID addressing header is omitted.

5.2.1 Summary

Uses WS-Addressing Echo contract as described in the WS-Addressing test suite RECEIVER is configured with WS-Addressing 1.0 and a SOAP 1.2 binding. SENDER sends Echo request message with a missing MessageID header. RECEIVER must return an appropriate addressing fault.

5.2.2 Message Exchange

SENDER sends a valid two-way message with no MessageID header. RECEIVER responds over the HTTP response with an appropriate SOAP fault.

5.2.2.1 Sample SENDER Request

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/echoIn</a:Action>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressing10</a:To>
  </s:Header>
  <s:Body>
    <echoIn xmlns="http://example.org/echo">test1291</echoIn>
  </s:Body>
</s:Envelope>
```

5.2.2.2 Sample RECEIVER Response

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
<a:Action>http://www.w3.org/2005/08/addressing/fault</a:Action>
  <a:MessageID>urn:uuid:436906E5-D724-42A8-BDE1-
A963B01676A1</a:MessageID>
</s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
```

```
          <s:Value>a:MessageAddressingHeaderRequired</s:Value>
        </s:Subcode>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en">A header representing a Message
Addressing Property is not valid and the message cannot be
processed</s:Text>
     </s:Reason>
     <s:Detail>
      <a:ProblemHeaderQName>a:MessageID</a:ProblemHeaderQName>
     </s:Detail>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

5.3 Test 1292

For RECEIVERs that do not support addressing, ensure that any request containing an addressing header with MustUnderstand attribute with value "1" will solicit a fault in the HTTP response.

5.3.1 Summary

RECEIVER implements the Signature contract (see section 5 below) with a document-literal binding, operation Sign3 (the only operation used in this test) takes no parameters. RECEIVER is configured with a SOAP 1.2 binding and no addressing. SENDER sends Echo request message with a To header containing a MustUnderstand attribute with value "1". RECEIVER responds with a MustUnderstand fault. This message exchange is repeated three additional times with the FaultTo, ReplyTo, and MessageID headers containing a MustUnderstand attribute with value "1".

5.3.2 Message Exchange

1. SENDER sends a valid two-way message with To header containing a MustUnderstand attribute with value "1". RECEIVER responds with a MustUnderstand fault over the HTTP response.
2. SENDER sends a valid two-way message with a FaultTo header containing a MustUnderstand attribute with value "1". RECEIVER responds with a MustUnderstand fault over the HTTP response.
3. SENDER sends a valid two-way message with a ReplyTo header containing a MustUnderstand attribute with value "1". RECEIVER responds with a MustUnderstand fault over the HTTP response.
4. SENDER sends a valid two-way message with a MessageID header containing a MustUnderstand attribute with value "1". RECEIVER responds with a MustUnderstand fault over the HTTP response.

5.3.2.1 Sample SENDER Request 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:MessageID>urn:uuid:cb2f6bb3-c2ac-4cc9-b1a7-6df86023dc7a</a:MessageID>
```

```
<a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addre
ss></a:ReplyTo>

<a:FaultTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addre
ss></a:FaultTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>
```

5.3.2.2 Sample RECEIVER Response 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
     <s:NotUnderstood qname="a:To"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:Action"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:MustUnderstand</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">One or more mandatory header blocks not
understood by the receiver.</s:Text>
      </s:Reason>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

5.3.2.3 Sample SENDER Request 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:MessageID>urn:uuid:1d37ec5c-65a0-4dc6-a72a-b47a15743754</a:MessageID>

<a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addre
ss></a:ReplyTo>
    <a:FaultTo
s:mustUnderstand="1"><a:Address>http://www.w3.org/2005/08/addressing/anonymou
s</a:Address></a:FaultTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
     </s:Envelope>
```

5.3.2.4 Sample RECEIVER Response 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
     <s:NotUnderstood qname="a:FaultTo"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:To"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:Action"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:MustUnderstand</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">One or more mandatory header blocks not
understood by the receiver.</s:Text>
      </s:Reason>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

5.3.2.5 Sample SENDER Request 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:MessageID>urn:uuid:00144363-c33f-4c4d-8f3e-b108b0e77695</a:MessageID>
    <a:ReplyTo
s:mustUnderstand="1"><a:Address>http://www.w3.org/2005/08/addressing/anonymou
s</a:Address></a:ReplyTo>

<a:FaultTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addre
ss></a:FaultTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>
```

5.3.2.6 Sample RECEIVER Response 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
     <s:NotUnderstood qname="a:ReplyTo"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:To"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:Action"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
  </s:Header>
  <s:Body>
    <s:Fault>
```

```
      <s:Code>
        <s:Value>s:MustUnderstand</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">One or more mandatory header blocks not
understood by the receiver.</s:Text>
      </s:Reason>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

5.3.2.7 Sample SENDER Request 4

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:MessageID s:mustUnderstand="1">urn:uuid:2bf961cb-e56c-4b0e-82db-
a64a9cf5a68f</a:MessageID>

<a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addre
ss></a:ReplyTo>

<a:FaultTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Addre
ss></a:FaultTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>
```

5.3.2.8 Sample RECEIVER Response 4

```
<s:Envelope xmlns:a=http://www.w3.org/2005/08/addressing”
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
     <s:NotUnderstood qname="a:MessageID"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:To"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
     <s:NotUnderstood qname="a:Action"
xmlns:wsa="http://www.w3.org/2005/08/addressing" />
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:MustUnderstand</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">One or more mandatory header blocks not
understood by the receiver.</s:Text>
      </s:Reason>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

5.4 Test 1293

Ensure that the recipient of a message with an invalid envelope namespace returns a VersionMismatch fault on the HTTP response.

5.4.1 Summary

RECEIVER implements the Echo contract as described in the WS-Addressing test suite using SOAP 1.2. SENDER sends a message with an invalid envelope namespace. RECEIVER returns a VersionMismatch fault on the HTTP response.

5.4.2 Message Exchange

Service implements the Echo contract as described in the WS-Addressing test suite using SOAP 1.2.

1. SENDER sends a message with an invalid envelope namespace.

2. RECEIVER returns a VersionMismatch fault on the HTTP response.

5.4.2.1 Sample SENDER Request

```
<s:Envelope xmlns="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/ysoap-envelopey">
  <s:Header>
    <Action d3p1:mustUnderstand="1"
xmlns:d3p1="http://www.w3.org/2003/05/soap-
envelope">http://www.example.org/action/echoIn</Action>
    <FaultTo>
      <Address>http://www.example.org/fault/</Address>
    </FaultTo>
    <MessageID>urn:uuid:6fb643f1-a11e-4806-b82b-c6df479a851e</MessageID>
    <ReplyTo>
      <Address>http://www.example.org/fault</Address>
    </ReplyTo>
    <To d3p1:mustUnderstand="1" xmlns:d3p1="http://www.w3.org/2003/05/soap-
envelope">http://www.example.org/WSAddressing10</To>
  </s:Header>
  <s:Body>
    <echoIn i:nil="true" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://example.org/echo" />
  </s:Body>
</s:Envelope>
```

5.4.2.2 Sample RECEIVER Response

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
     <s:Upgrade>
       <s:SupportedEnvelope qname="s:Envelope" />
     </s:Upgrade>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:VersionMismatch</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">Version Mismatch</s:Text>
```

```
      </s:Reason>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

5.5 Test 1294

Application fault sent to a non-anonymous FaultTo address.

5.5.1 Summary

RECEIVER implements the echo contract as described in the WS-Addressing test suite using SOAP 1.2. SENDER sends a message containing the string "fault" in the body, using a FaultTo and ReplyTo address that is not anonymous and not None. Service responds with a fault in a new request to the given FaultTo address.

5.5.2 Message Exchange

1. SENDER sends message to Echo operation on RECEIVER. The string "fault" is contained in the request body. The message contains a ReplyTo and FaultTo addresses that are not anonymous and not None.
2. RECEIVER responds with a 200 OK or 202 Accepted over the HTTP response
3. RECEIVER sends a fault over a new HTTP request to the FaultTo address specified in the original request.
4. SENDER responds with a 200 OK or 202 Accepted over the HTTP response

5.5.2.1 Sample SENDER Request (step 1)

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/echoIn</a:Action>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressing10</a:To>
    <a:MessageID>urn:uuid:b7ce162e-96d6-480e-9ccc-081c61d86795</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.example.org/reply</a:Address>
    </a:ReplyTo>
    <a:FaultTo>
      <a:Address>http://www.example.org/reply</a:Address>
    </a:FaultTo>
  </s:Header>
  <s:Body>
    <echoIn xmlns="http://example.org/echo">faulttest1294</echoIn>
  </s:Body>
</s:Envelope>
```

5.5.2.2 Sample RECEIVER Response (step 3)

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
```

```
    <a:Action
s:mustUnderstand="1">http://www.w3.org/2005/08/addressing/soap/fault</a:Actio
n>
    <a:RelatesTo>urn:uuid:b7ce162e-96d6-480e-9ccc-081c61d86795</a:RelatesTo>
    <a:To s:mustUnderstand="1"> http://www.example.org/reply</a:To>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
        <s:Subcode>
          <s:Value
xmlns:a="http://example.org/echo">a:EmptyEchoString</s:Value>
        </s:Subcode>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">The supplied string was of zero
length.</s:Text>
      </s:Reason>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

6 Signature Tests

In these tests, we verify compliance with the rules for operation signatures and the corresponding rules for WSDL descriptions for document-literal and rpc-literal bindings.

- RECEIVERs correctly dispatch messages between operations with identical top-level body elements but different actions. Operations with identical top-level body elements but different actions result in unique operation signatures.
- RECEIVERs correctly dispatch messages between operations with no body elements but different actions. Operations with no body elements but different actions result in unique operation signatures.
- SENDERs correctly order operation parameters (or children/grandchildren of the soap body element) based on WSDL description. Operation parameter order (the order of children/grandchildren of the soap body element) is correctly communicated in WSDL.

Additionally the tests verify that headers supplied as parameters for particular operations and potential soap faults are correctly communicated through WSDL.

The WSDL description for these tests, including schema, is provided in the WSDL directory of this package (WSDL\AddressingAndSignature\WSAddressingSignature.wsdl).

6.1 Contracts and RECEIVER Behavior

Each WSDL portType in these tests exposes six operations. Two operations specify a single parameter (of type string), each using a different action. Two operations specify no parameters. One operation specifies two parameters of type string. The final operation defines both a SOAP header and soap:body

element contained in the request and response messages and includes a SOAP fault in the WSDL description.

6.2 Test 1297

Test 1297 uses a document-literal binding with WS-Addressing 1.0 and SOAP 1.2. The first two operations have different actions, but define the same input soap body element of type string and return the same output body element of type string. The second two operations have different actions and specify no input SOAP body element and the same output body element of type string. The description for the fifth operation contains an empty parts attribute for the associated input and output messages. The sixth operation specifies two input body elements of type string and an output body element of type string. The seventh operation specifies an input header and body element, both of type string, and an output header and body element, also of type string. It also specifies a SOAP fault that the operation may throw.

The SENDER sends a message to each operation. The RECEIVER responds with a string containing the operation name and passed-in parameters. The SENDER sends a seventh message to the last operation with the string "fault" in the header or body parameter, the RECEIVER responds with the fault specified in WSDL.

6.2.1 Message Exchange

1. SENDER sends a message to operation 1, RECEIVER responds with a message containing operation name and the passed parameter over the HTTP response.
2. SENDER sends a message to operation 2, RECEIVER responds with a message containing operation name and the passed parameter over the HTTP response.
3. SENDER sends a message to operation 3, RECEIVER responds with a message containing operation name.
4. SENDER sends a message to operation 4, RECEIVER responds with a message containing operation name over the HTTP response.
5. SENDER sends a message to operation 5, RECEIVER responds with the specified output message over the http response.
6. SENDER sends a message to operation 6, RECEIVER responds with a message containing operation name and the two passed parameters over the HTTP response.
7. SENDER sends a message to operation 7. RECEIVER creates a string containing operation name and the parameter passed in the header followed by the parameter passed in the body, returning a message over the HTTP response that contains this string in the header and body elements specified in WSDL.
8. SENDER sends a message to operation 7 with the string "fault" contaminated in the header or body parameter, RECEIVER responds with the fault specified in the operation's WSDL description over the HTTP response.

6.2.1.1 Sample SENDER Request 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
```

```
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureIn</a:Action>
    <a:MessageID>urn:uuid:b99acebe-67fa-4d21-8ee7-82f62648860d</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
  </s:Header>
  <s:Body>
    <SignatureIn xmlns="http://example.org/signature">Hello</SignatureIn>
  </s:Body>
</s:Envelope>
```

6.2.1.2 Sample SENDER Request 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureInAlternate</a:Action
>
    <a:MessageID>urn:uuid:9d51c3c1-029d-4ee5-9b47-f2d681ccc661</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1"> http://www.example.org/WSAddressingSignature
</a:To>
  </s:Header>
  <s:Body>
    <SignatureIn xmlns="http://example.org/signature">Hello</SignatureIn>
  </s:Body>
</s:Envelope>
```

6.2.1.3 Sample SENDER Request 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureInEmpty</a:Action>
    <a:MessageID>urn:uuid:7358118f-e7a5-45aa-ab6c-62b41c7c6d59</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1"> http://www.example.org/WSAddressingSignature
</a:To>
  </s:Header>
  <s:Body />
</s:Envelope>
```

6.2.1.4 Sample SENDER Request 4

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureInAlternateEmpty</a:A
ction>
```

```
    <a:MessageID>urn:uuid:58274114-a369-4802-9bd1-3562787346bd</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To
s:mustUnderstand="1">http://www.example.org/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap12</a:To>
  </s:Header>
  <s:Body />
</s:Envelope>
```

6.2.1.5 Sample SENDER Request 5

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureEmptyPartIn</a:Ac
tion>
    <a:MessageID>urn:uuid:7358118f-e7a5-45aa-ab6c-62b41c7c6d59</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1
http://www.example.org/WSAddressingCR_Service_WCF/WSAddressingSignature.svc/S
oap12</a:To>
  </s:Header>
  <s:Body />
</s:Envelope>
```

6.2.1.6 Sample SENDER Request 6

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureInMultipart</a:Action
>
    <a:MessageID>urn:uuid:468c48dd-a086-4d31-9d47-45a6f08a4d0e</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
  </s:Header>
  <s:Body>
    <SignatureInRequest xmlns="http://example.org/signature">
      <SignatureInFirstPart>Hello</SignatureInFirstPart>
      <SignatureInSecondPart>World</SignatureInSecondPart>
    </SignatureInRequest>
  </s:Body>
</s:Envelope>
```

6.2.1.7 Sample SENDER Request 7

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
```

```
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureHeaderIn</a:Action>
    <h:SignatureInHeader
xmlns:h="http://example.org/signature">Hello</h:SignatureInHeader>
    <a:MessageID>urn:uuid:7cf2294c-abb8-406d-bf9c-4a66f2f817d4</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/WSAddressingSignature
</a:To>
  </s:Header>
  <s:Body>
    <SignatureInHeaderMember
xmlns="http://example.org/signature">World</SignatureInHeaderMember>
  </s:Body>
</s:Envelope>
```

6.2.1.8 Sample SENDER Request 8

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureHeaderIn</a:Action>
    <h:SignatureInHeader
xmlns:h="http://example.org/signature">Fault</h:SignatureInHeader>
    <a:MessageID>urn:uuid:a91c1670-2d0a-4b24-895d-83e59dffa5e8</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1"> http://www.example.org/WSAddressingSignature
</a:To>
  </s:Header>
  <s:Body>
    <SignatureInHeaderMember
xmlns="http://example.org/signature">World</SignatureInHeaderMember>
  </s:Body>
</s:Envelope>
```

6.2.1.9 Sample RECEIVER Response 1

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureOut</a:Action>
    <a:RelatesTo>urn:uuid:b99acebe-67fa-4d21-8ee7-82f62648860d</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://example.org/signature">Operation=Sign1,
Parameter=Hello</SignatureOut>
  </s:Body>
</s:Envelope>
```

6.2.1.10 Sample RECEIVER Response 2

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
```

```
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureOutAlternate</a:Actio
n>
    <a:RelatesTo>urn:uuid:9d51c3c1-029d-4ee5-9b47-f2d681ccc661</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://example.org/signature">Operation=Sign2,
Parameter=Hello</SignatureOut>
  </s:Body>
</s:Envelope>
```

6.2.1.11 Sample RECEIVER Response 3

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureOutEmpty</a:Action>
    <a:RelatesTo>urn:uuid:7358118f-e7a5-45aa-ab6c-62b41c7c6d59</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://example.org/signature">Operation=Sign3
</SignatureOut>
  </s:Body>
</s:Envelope>
```

6.2.1.12 Sample RECEIVER Response 4

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureOutAlternateEmpty</a:
Action>
    <a:RelatesTo>urn:uuid:58274114-a369-4802-9bd1-3562787346bd</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://example.org/signature">Operation=Sign4
</SignatureOut>
  </s:Body>
</s:Envelope>
```

6.2.1.13 Sample RECEIVER Response 5

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureEmptyPartOut</a:A
ction>
    <a:RelatesTo>urn:uuid:7358118f-e7a5-45aa-ab6c-62b41c7c6d59</a:RelatesTo>
  </s:Header>
  <s:Body />
</s:Envelope>
```

6.2.1.14 Sample RECEIVER Response 6

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
```

```
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureOutMultipart</a:Actio
n>
    <a:RelatesTo>urn:uuid:468c48dd-a086-4d31-9d47-45a6f08a4d0e</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://example.org/signature">Operation=Sign6,
Parameter=Hello, Parameter=World</SignatureOut>
  </s:Body>
</s:Envelope>
```

6.2.1.15 Sample RECEIVER Response 7

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureHeaderReply</a:Action
>
    <h:SignatureOutHeader
xmlns:h="http://example.org/signature">Operation=Sign7, Parameter=Hello,
Parameter=World</h:SignatureOutHeader>
    <a:RelatesTo>urn:uuid:7cf2294c-abb8-406d-bf9c-4a66f2f817d4</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOutHeaderMember
xmlns="http://example.org/signature">Operation=Sign7, Parameter=Hello,
Parameter=World</SignatureOutHeaderMember>
  </s:Body>
</s:Envelope>
```

6.2.1.16 Sample RECEIVER Response 8

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureHeaderFault</a:Action
>
    <a:RelatesTo>urn:uuid:a91c1670-2d0a-4b24-895d-83e59dffa5e8</a:RelatesTo>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US">The creator of this fault did not specify a
Reason.</s:Text>
      </s:Reason>
      <s:Detail>
        <SignatureHeaderFaultContract xmlns="http://example.org/signature"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <FaultAdditionalText>Fault: Fault, World</FaultAdditionalText>
        </SignatureHeaderFaultContract>
      </s:Detail>
    </s:Fault>
```

```
  </s:Body>
</s:Envelope>
```

6.3 Test 1298

Test 1298 uses an rpc-literal binding with WS-Addressing 1.0 over SOAP 1.2. The first operation has no input parameters and an output parameter of type string. The description for the second operation uses input and output messages with empty parts attributes. The third operation specifies two input parameters of type string, and an output parameter, also of type string. It also specifies a SOAP fault that the operation may throw.

The SENDER sends a message to each operation. The RECEIVER responds with a string containing the operation name and passed-in parameters. The SENDER sends a fourth message to the third operation with the string "fault" in the header or body parameter, the RECEIVER responds with the fault specified in WSDL.

Message Exchange

1. SENDER sends a message to operation 1, RECEIVER responds with a message containing operation name over the http response.
2. SENDER sends a message to operation 2, RECEIVER responds with the specified output message over the HTTP response.
3. SENDER sends a request to operation 3. RECEIVER responds with a message containing the operation name and two passed parameters over the HTTP response.
4. SENDER sends a message to operation 3 with the string "fault" contained in one of the two parameters, RECEIVER responds with the fault specified in the operation's WSDL description over the HTTP response.
5. SENDER sends a request to operation 3, using an 'action' parameter in the HTTP content-type header that matches the SOAPAction for operation 6. RECEIVER responds with a message containing the operation name and two passed parameters over the HTTP response.

6.3.1.1 Sample SENDER Request 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureRpcInEmpty</a:Act
ion>
    <a:MessageID>urn:uuid:5347f94d-1d4c-4926-a981-ec086c54cf92</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/Soap12/Rpc</a:To>
  </s:Header>
  <s:Body>
    <Sign1 xmlns="http://www.example.org/signature" />
```

```
  </s:Body>
</s:Envelope>
```

6.3.1.2 Sample SENDER Request 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureEmptyPartRpc</a:A
ction>
    <a:MessageID>urn:uuid:7358118f-e7a5-45aa-ab6c-62b41c7c6d59</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/Soap12/Rpc</a:To>
  </s:Header>
  <s:Body>
    <Sign2 xmlns=http://example.org />
  </s:Body>
</s:Envelope>
```

6.3.1.3 Sample SENDER Request 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderRpcIn</a:Ac
tion>
    <a:MessageID>urn:uuid:e425684f-adcd-4701-a333-0ae479600ee2</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/Soap12/Rpc</a:To>
  </s:Header>
  <s:Body>
    <Sign3 xmlns="http://www.example.org/signature">
      <parameter1 xmlns="">Hello</parameter1>
      <parameter2 xmlns="">World</parameter2>
    </Sign3>
  </s:Body>
</s:Envelope>
```

6.3.1.4 Sample SENDER Request 4

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderRpcIn</a:Ac
tion>
    <a:MessageID>urn:uuid:4c6d6ae8-8cb2-4870-b5b4-cfaecdf66bdb</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/Soap12/Rpc</a:To>
  </s:Header>
  <s:Body>
```

```
    <Sign3 xmlns="http://www.example.org/signature">
      <parameter1 xmlns="">Fault</parameter1>
      <parameter2 xmlns="">World</parameter2>
    </Sign3>
  </s:Body>
</s:Envelope>
```

6.3.1.5 Sample SENDER Request 5 (R2744)

```
<wsil:testLog xmlns:wsil="http://www.ws-i.org/testing/2008/02/log/">
<wsil:messageLog>
<wsil:message type="request" id="1" conversation="1">
<wsil:httpHeaders>
  <wsil:requestLine>POST /Soap12/Rpc HTTP/1.1</wsil:requestLine>
  <wsil:contentTypeHeader type="application" subtype="soap+xml">
    <wsil:parameter value="utf-8" key="charset"/>
    <wsil:parameter
value="http://www.example.org/action/SignatureHeaderRpcIn" key="action"/>
  </wsil:contentTypeHeader>
</wsil:httpHeaders>
<wsil:messageContents encoding="utf-8" validXml="true" xmlVersion="1.0"
containsProcessingInstructions="false" containsDTD="false">
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderRpcIn</a:Ac
tion>
    <a:MessageID>urn:uuid:e425684f-adcd-4701-a333-0ae479600ee8</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://www.example.org/Soap12/Rpc</a:To>
  </s:Header>
  <s:Body>
    <Sign3 xmlns="http://www.example.org/signature">
      <parameter1 xmlns="">HelloAgain</parameter1>
      <parameter2 xmlns="">World</parameter2>
    </Sign3>
  </s:Body>
</s:Envelope>
</wsil:messageContents>
</wsil:message>
</wsil:messageLog>
</wsil:testLog>
```

6.3.1.6 Sample RECEIVER Response 1

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureRpcOutEmpty</a:Ac
tion>
    <a:RelatesTo>urn:uuid:5347f94d-1d4c-4926-a981-ec086c54cf92</a:RelatesTo>
```

```
  </s:Header>
  <s:Body>
    <Sign1Response xmlns="http://www.example.org/signature">
      <Sign1Result xmlns="">Sign1</Sign1Result>
    </Sign1Response>
  </s:Body>
</s:Envelope>
```

6.3.1.7 Sample RECEIVER Response 2

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">http://www.example.org/action/
SignatureEmptyPartRpcReply</a:Action>
    <a:RelatesTo>urn:uuid:7358118f-e7a5-45aa-ab6c-62b41c7c6d59</a:RelatesTo>
  </s:Header>
  <s:Body>
    <Sign2 xmlns="http://example.org" />
  </s:Body>
</s:Envelope>
```

6.3.1.8 Sample RECEIVER Response 3

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderRpcReply</a
:Action>
    <a:RelatesTo>urn:uuid:e425684f-adcd-4701-a333-0ae479600ee2</a:RelatesTo>
  </s:Header>
  <s:Body>
    <Sign3Response xmlns="http://www.example.org/signature">
      <Sign3Result xmlns="">Sign3, Parameter=Hello,
Parameter=World</Sign3Result>
    </Sign3Response>
  </s:Body>
</s:Envelope>
```

6.3.1.9 Sample RECEIVER Response 4

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureHeaderFaultRpc</a:Act
ion>
    <a:RelatesTo>urn:uuid:36aed9ca-2a48-46a1-b515-051a98a7dbe9</a:RelatesTo>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:Code>
        <s:Value>s:Sender</s:Value>
      </s:Code>
      <s:Reason>
        <s:Text xml:lang="en-US"><fault reason></s:Text>
      </s:Reason>
      <s:Detail>
```

```
        <SignatureHeaderFaultContractRpc xmlns="http://example.org/signature"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <FaultAdditionalTextRpc>Fault: Fault,
World</FaultAdditionalTextRpc>
        </SignatureHeaderFaultContractRpc>
      </s:Detail>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

6.3.1.10 Sample RECEIVER Response 5 (2744)

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://example.org/action/SignatureHeaderRpcReply</a:Act
ion>
    <a:RelatesTo>urn:uuid:e425684f-adcd-4701-a333-0ae479600ee8</a:RelatesTo>
  </s:Header>
  <s:Body>
    <Sign3Response xmlns="http://www.example.org/signature">
      <Sign3Result xmlns="">Operation=Sign3, Parameter=HelloAgain,
Parameter=World</Sign3Result>
    </Sign3Response>
  </s:Body>
</s:Envelope>
```

7 Acknowledgements

The following experts have contributed to the development and submission of the test scenarios:

Mark Cowlishaw, Microsoft

Asad Jawahar, Microsoft

Vipul Modi, Microsoft

Kirill Gavrylyuk, Microsoft

Monica J. Martin, Microsoft

Asir Vedamuthu, Microsoft

8 Revision History

Date	Revisions	Comments
26 August 2008	Add revisions as documented or raised during the F2F in Sunnyvale CA of Basic Profile WG.	Revisions integrated.
7 September 2008	Integrated further changes from	Revisions integrated.

	WG meeting 2 September 2008 and additional package files added (sample WSDL references).	
9 September 2008	Added new scenarios to cover R2213 and R2214. Made minor editorial corrections such as correct test log format for one test scenario.	Revisions integrated.
24 October 2008	Corrected test 2930 message exchange action	Revisions integrated.
4 November 2008	Changed operation names for BaseDataTypes tests to correspond with XML Schema types.	Revisions integrated.
18 November 2008	Incorporated changes from Basic Profile face to face testing 11/11-11/13/2008.	Revisions integrated.
24 November 2008	Approved as Working Group Draft.	Revisions integrated.