

## Working Group Draft

**2008-11-24**

*This version:*

Same

*Latest version:*

Same

*Last version:*

<http://tinyurl.com/66nmjv> (11/18/2008)

*Editors:*

Robert Chumbley, IBM

Monica J. Martin, Microsoft

*Administrative contact:*

[secretary@ws-i.org](mailto:secretary@ws-i.org)

Copyright © 2002-2008 by [The Web Services-Interoperability Organization](#) (WS-I) and Certain of its Members. All Rights Reserved.

### **1.1 Abstract**

This document provides a set of scenarios for testing interoperability under Basic Profile 1.2 focused on SOAP, WSDL, 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 1.1. The WS-Addressing Test Suite is used in addition to other scenarios to effectively test Basic Profile 1.2.

### **1.2 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.

### **1.3 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.

#### **1.4 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](mailto:wsbasic_comment@ws-i.org).

## Table of Contents

1.1	Abstract.....	1
1.2	Status of this Document.....	1
1.3	Notice.....	1
1.4	Feedback.....	2
1	Schema and WSDL Tests.....	5
1.1	Test Procedure.....	6
1.2	Flow of Testing Schema and WSDL Scenarios.....	6
1.3	Base Data Types.....	6
1.3.1	RetAnyType Test Description.....	7
1.4	Complex Data Types.....	7
1.5	Sample Messages for Schema and WSDL Tests.....	8
1.6	Test 0801 (RetString UTF-16).....	8
1.6.1	Sample SENDER Request Message.....	8
1.6.2	Sample RECEIVER Response Message.....	9
2	New WS-Addressing Tests.....	9
2.1	Test 1191.....	9
2.1.1	Summary.....	9
2.1.2	Message Exchange.....	9
2.1.2.1	Sample SENDER Request.....	9
2.1.2.2	Sample RECEIVER Response.....	10
2.2	Test 1192.....	10
2.2.1	Summary.....	10
2.2.2	Message Exchange.....	10
2.2.2.1	Sample SENDER Request 1.....	11
2.2.2.2	Sample RECEIVER Response 1.....	11
2.2.2.3	Sample SENDER Request 2.....	11
2.2.2.4	Sample RECEIVER Response 2.....	12
2.2.2.5	Sample SENDER Request 3.....	12

2.2.2.6	Sample RECEIVER Response 3 .....	12
2.2.2.7	Sample SENDER Request 4 .....	12
2.2.2.8	Sample RECEIVER Response 4 .....	13
2.3	Test 1193 .....	13
2.3.1	Summary .....	13
2.3.2	Message Exchange .....	13
2.3.2.1	Sample SENDER Request .....	13
2.3.2.2	Sample RECEIVER Response .....	14
2.4	Test 1194 .....	14
2.4.1	Summary .....	14
2.4.2	Message Exchange .....	14
2.4.2.1	Sample SENDER Request (step 1) .....	15
2.4.2.2	Sample RECEIVER Response (step 3) .....	15
3	Signature Tests .....	15
3.1	Contracts and RECEIVER Behavior .....	16
3.2	Test 1197 .....	16
3.2.1	Message Exchange .....	16
3.2.1.1	Sample SENDER Request 1 .....	17
3.2.1.2	Sample SENDER Request 2 .....	17
3.2.1.3	Sample SENDER Request 3 .....	18
3.2.1.4	Sample SENDER Request 4 .....	18
3.2.1.5	Sample SENDER Request 5 .....	18
3.2.1.6	Sample SENDER Request 6 .....	19
3.2.1.7	Sample SENDER Request 7 .....	19
3.2.1.8	Sample SENDER Request 8 .....	19
3.2.1.9	Sample RECEIVER Response 1 .....	20
3.2.1.10	Sample RECEIVER Response 2 .....	20
3.2.1.11	Sample RECEIVER Response 3 .....	20
3.2.1.12	Sample RECEIVER Response 4 .....	21
3.2.1.13	Sample RECEIVER Response 5 .....	21
3.2.1.14	Sample RECEIVER Response 6 .....	21
3.2.1.15	Sample RECEIVER Response 7 .....	21

3.2.1.16	Sample RECEIVER Response 8.....	22
3.3	Test 1198.....	22
3.3.1	Message Exchange .....	22
3.3.1.1	Sample SENDER Request 1.....	23
3.3.1.2	Sample SENDER Request 2.....	23
3.3.1.3	Sample SENDER Request 3.....	23
3.3.1.4	Sample SENDER Request 4.....	24
3.3.1.5	Sample RECEIVER Response 1.....	24
3.3.1.6	Sample RECEIVER Response 2.....	25
3.3.1.7	Sample RECEIVER Response 3.....	25
3.3.1.8	Sample RECEIVER Response 4.....	25
4	Acknowledgements .....	26
5	Revision History .....	26

## Introduction

This document provides a set of scenarios for testing interoperability under Basic Profile 1.2. 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 1.2. 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
- [Web Services Addressing 1.0 Test Suite](#)
  - All Required SOAP 1.1 tests (test11xx).
- Additional tests to effectively test Basic Profile 1.2 requirements.

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

## 1 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 and WSDL are provided with these scenarios.

The scenarios are further categorized based on the data type. The basic data type scenarios test 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.

## 1.1 Test Procedure

All scenarios have two participants, SENDER and RECEIVER, consistent with the roles defined in Basic Profile 1.2 (See [Section 2.2 Conformance Targets](#)). All of the scenarios are "" 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.

## 1.2 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.

## 1.3 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	<a href="#">Click here</a>	<a href="#">Click here</a>
RetUnsignedByte	xs:unsignedByte	<a href="#">Click here</a>	<a href="#">Click here</a>
RetByte	xs:byte	<a href="#">Click here</a>	<a href="#">Click here</a>
RetBase64Binary	xs:base64Binary	<a href="#">Click here</a>	<a href="#">Click here</a>
RetDecimal	xs:decimal	<a href="#">Click here</a>	<a href="#">Click here</a>
RetFloat	xs:float	<a href="#">Click here</a>	<a href="#">Click here</a>
RetDouble	xs:double	<a href="#">Click here</a>	<a href="#">Click here</a>

RetInt	xs:int	<a href="#">Click here</a>	<a href="#">Click here</a>
RetShort	xs:short	<a href="#">Click here</a>	<a href="#">Click here</a>
RetLong	xs:long	<a href="#">Click here</a>	<a href="#">Click here</a>
RetAnyType* (see explanation below)	xs:anyType	<a href="#">Click here</a>	<a href="#">Click here</a>
RetUnsignedInt	xs:unsignedInt	<a href="#">Click here</a>	<a href="#">Click here</a>
RetUnsignedShort	xs:unsignedShort	<a href="#">Click here</a>	<a href="#">Click here</a>
RetUnsignedLong	xs:unsignedLong	<a href="#">Click here</a>	<a href="#">Click here</a>
RetString	xs:string	<a href="#">Click here</a>	<a href="#">Click here</a>
RetAnyUri	xs:anyURI	<a href="#">Click here</a>	<a href="#">Click here</a>
RetDateTime	xs:dateTime	<a href="#">Click here</a>	<a href="#">Click here</a>
RetDuration	xs:duration	<a href="#">Click here</a>	<a href="#">Click here</a>
RetQName	xs:QName	<a href="#">Click here</a>	<a href="#">Click here</a>

### 1.3.1 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.

## 1.4 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	<a href="#">Click here</a>	<a href="#">Click here</a>
RetArrayInt1D	<a href="#">Click here</a>	<a href="#">Click here</a>

## 1.5 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.

Sample WSDL for these services are provided in the accompanying zip file (WSDL\SchemaAndWSDL).

## 1.6 Test 0801 (RetString UTF-16)

The description for this scenario is encoded using UTF-16 text encoding. This test uses a document-literal binding with SOAP 1.1. 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.

### 1.6.1 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 /BaseDataTypesDocLitWUtf16.svc
HTTP/1.1</wsil:requestLine>
  <wsil:contentTypeHeader type="text" subtype="xml">
    <wsil:parameter value="utf-16" key="charset"/>
  </wsil:contentTypeHeader>
  <wsil:httpHeader value="http://tempuri.org/IBaseDataTypesDocLitW/RetString"
key="SOAPAction"/>
</wsil:httpHeaders>
<wsil:messageContents encoding="utf-16" validXml="true" xmlVersion="1.0"
containsProcessingInstructions="false" containsDTD="false">
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://tempur
i.org/IBaseDataTypesDocLitW/RetString</Action>
    <To s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://exampl
e.org/BaseDataTypesDocLitWUtf16</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>
```

## 1.6.2 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="text" subtype="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://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <RetStringResponse xmlns="http://tempuri.org/">
      <RetStringResult>Hello World</RetStringResult>
    </RetStringResponse>
  </s:Body>
</s:Envelope>
</wsil:messageContents>
</wsil:message>
</wsil:messageLog>
</wsil:testLog>
```

## 2 New WS-Addressing Tests

### 2.1 Test 1191

Test for correct response when MessageID addressing header is omitted.

#### 2.1.1 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 1.1 binding. SENDER sends Echo request message with a missing MessageID header. RECEIVER must return an appropriate addressing fault.

#### 2.1.2 Message Exchange

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

##### 2.1.2.1 Sample SENDER Request

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.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/WSAddressingCR_Service_WCF/WSAddr
essing10.svc/Soap11</a:To>
  </s:Header>
  <s:Body>
    <echoIn xmlns="http://www.example.org/echo">test1191</echoIn>
  </s:Body>
</s:Envelope>

```

### 2.1.2.2 Sample RECEIVER Response

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/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/fault</a:Action>
  </s:Header>
  <s:Body>
    <s:Fault>
      <s:faultCode>a:MessageAddressingHeaderRequired
      </s:faultCode>
      <s:faultString xml:lang="en">A header representing a Message
Addressing Property is not valid and the message cannot be processed
      </s:faultString>
    </s:Fault>
  </s:Body>
</s:Envelope>

```

## 2.2 Test 1192

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

### 2.2.1 Summary

RECEIVER implements the Signature contract (see below) with a document-literal binding, operation Sign3 (the only operation used in this test) takes no parameters. RECEIVER is configured with a SOAP 1.1 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”.

### 2.2.2 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.

### 2.2.2.1 Sample SENDER Request 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:MessageID>urn:uuid:97bf9cb1-7e2c-4bf1-9354-35df4a7be133</a:MessageID>
    <a:ReplyTo>http://www.w3.org/2005/08/addressing/envelope</a:ReplyTo>
    <a:FaultTo>http://www.example.com/</a:FaultTo>
    <a:To
s:mustUnderstand="1">http://www.example.org/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/None</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>
```

### 2.2.2.2 Sample RECEIVER Response 1

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <s:Fault>
      <faultcode>s:MustUnderstand</faultcode>
      <faultstring xml:lang="en-US">The header 'To' from the namespace
'http://www.w3.org/2005/08/addressing' was not understood by the recipient of
this message, causing the message to not be processed. This error typically
indicates that the sender of this message has enabled a communication
protocol that the receiver cannot process. Please ensure that the
configuration of the SENDER's binding is consistent with the RECEIVER's
binding. </faultstring>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

### 2.2.2.3 Sample SENDER Request 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:MessageID>urn:uuid:ea978599-4afe-4257-8ade-bb7dd8e3d58b</a:MessageID>
    <a:ReplyTo>http://www.w3.org/2005/08/addressing/envelope</a:ReplyTo>
    <a:FaultTo s:mustUnderstand="1">http://www.example.com/</a:FaultTo>
    <a:To
s:mustUnderstand="1">http://www.example.org/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/None</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>
```

#### 2.2.2.4 Sample RECEIVER Response 2

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <s:Fault>
      <faultcode>s:MustUnderstand</faultcode>
      <faultstring xml:lang="en-US">The header 'FaultTo' from the namespace
'http://www.w3.org/2005/08/addressing' was not understood by the recipient of
this message, causing the message to not be processed. This error typically
indicates that the sender of this message has enabled a communication
protocol that the receiver cannot process. Please ensure that the
configuration of the SENDER's binding is consistent with the RECEIVER's
binding. </faultstring>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

#### 2.2.2.5 Sample SENDER Request 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:MessageID>urn:uuid:9d74476d-4996-4360-b4e4-522fda9347f6</a:MessageID>
    <a:ReplyTo
s:mustUnderstand="1">http://www.w3.org/2005/08/addressing/envelope</a:ReplyTo
>
    <a:FaultTo>http://www.example.com/</a:FaultTo>
    <a:To
s:mustUnderstand="1">http://www.example.org/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/None</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>
```

#### 2.2.2.6 Sample RECEIVER Response 3

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <s:Fault>
      <faultcode>s:MustUnderstand</faultcode>
      <faultstring xml:lang="en-US">The header 'ReplyTo' from the namespace
'http://www.w3.org/2005/08/addressing' was not understood by the recipient of
this message, causing the message to not be processed. This error typically
indicates that the sender of this message has enabled a communication
protocol that the receiver cannot process. Please ensure that the
configuration of the SENDER's binding is consistent with the service's
binding. </faultstring>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

#### 2.2.2.7 Sample SENDER Request 4

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

```

    <a:MessageID s:mustUnderstand="1">urn:uuid:c5ef83eb-b12d-4f25-b512-
2bd326c90c8b</a:MessageID>
    <a:ReplyTo>http://www.w3.org/2005/08/addressing/envelope</a:ReplyTo>
    <a:FaultTo>http://www.example.com/</a:FaultTo>
    <a:To
s:mustUnderstand="1">http://www.example.org/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/None</a:To>
    <a:Action
s:mustUnderstand="1">http://www.example.org/signature/SignatureDocumentLitera
lNoAction/Sign3</a:Action>
  </s:Header>
  <s:Body />
</s:Envelope>

```

#### 2.2.2.8 Sample RECEIVER Response 4

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <s:Fault>
      <faultcode>s:MustUnderstand</faultcode>
      <faultstring xml:lang="en-US">The header 'MessageID' from the namespace
'http://www.w3.org/2005/08/addressing' was not understood by the recipient of
this message, causing the message to not be processed. This error typically
indicates that the sender of this message has enabled a communication
protocol that the receiver cannot process. Please ensure that the
configuration of the SENDER's binding is consistent with the RECEIVER's
binding. </faultstring>
    </s:Fault>
  </s:Body>
</s:Envelope>

```

## 2.3 Test 1193

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

### 2.3.1 Summary

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

### 2.3.2 Message Exchange

Service implements the Echo contract as described in the [WS-Addressing test suite](#) using SOAP 1.1.

1. SENDER sends a message with an invalid envelope namespace.
2. RECEIVER returns a VersionMismatch fault on the HTTP response.

#### 2.3.2.1 Sample SENDER Request

```

<s:Envelope xmlns="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/yenvelopey">
  <s:Header>

```

```

    <Action d3p1:mustUnderstand="1"
xmlns:d3p1="http://schemas.xmlsoap.org/soap/envelope/">http://www.example.org
/action/echoIn</Action>
    <FaultTo>
        <Address>http://www.example.com/</Address>
    </FaultTo>
    <MessageID>urn:uuid:8fe2f94c-2fca-415f-b152-270a7b1c7075</MessageID>
    <ReplyTo>
        <Address>http://www.w3.org/2005/08/addressing/anonymous</Address>
    </ReplyTo>
    <To d3p1:mustUnderstand="1"
xmlns:d3p1="http://schemas.xmlsoap.org/soap/envelope/">http://www.example.org
/WSAddressingCR_Service_WCF/WSAddressing10.svc/Soap12</To>
</s:Header>
<s:Body>
    <echoIn i:nil="true" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.example.org/echo" />
</s:Body>
</s:Envelope>

```

### 2.3.2.2 Sample RECEIVER Response

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Body>
        <s:Fault>
            <faultcode s:VersionMismatch</faultcode>
            <faultstring xml:lang="en-US"> Version Mismatch.</faultstring>
        </s:Fault>
    </s:Body>
</s:Envelope>

```

## 2.4 Test 1194

Application fault sent to a non-anonymous FaultTo address.

### 2.4.1 Summary

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

### 2.4.2 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

### 2.4.2.1 Sample SENDER Request (step 1)

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/echoIn</a:Action>
    <a:To
s:mustUnderstand="1">http://www.example.org/WSAddressingCR_Service_WCF/WSAddr
essing10.svc/Soap11/Duplex</a:To>
    <a:MessageID>urn:uuid:974c7e5e-45ec-4b85-b5dc-530e0df5e75a</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.example.org/reply</a:Address>
    </a:ReplyTo>
    <a:FaultTo>
      <a:Address>http://www.example.org/fault</a:Address>
    </a:FaultTo>
  </s:Header>
  <s:Body>
    <echoIn xmlns="http://www.example.org/echo">faulttest1194</echoIn>
  </s:Body>
</s:Envelope>
```

### 2.4.2.2 Sample RECEIVER Response (step 3)

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/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:974c7e5e-45ec-4b85-b5dc-530e0df5e75a</a:RelatesTo>
    <a:To s:mustUnderstand="1">http://www.example.org/fault</a:To>
  </s:Header>
  <s:Body>
    <s:Fault>
      <faultcode
xmlns:a="http://www.example.org/echo">a:EmptyEchoString</faultcode>
      <faultstring xml:lang="en-US">The supplied string was of zero
length.</faultstring>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

## 3 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 grandchildren of the SOAP body element) based on WSDL description. Operation parameter order (the order of 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).

### 3.1 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 varies over the test, but always contains a SOAP fault in the WSDL description.

### 3.2 Test 1197

Test 1197 uses a document-literal binding with WS-Addressing 1.0. 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.

#### 3.2.1 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.

### 3.2.1.1 Sample SENDER Request 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureIn</a:Action>
    <a:MessageID>urn:uuid:80763f6e-999e-44bd-9d62-e7878ee1eec6</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/Soap11</a:To>
  </s:Header>
  <s:Body>
    <SignatureIn xmlns="http://www.example.org/signature">Hello</SignatureIn>
  </s:Body>
</s:Envelope>
```

### 3.2.1.2 Sample SENDER Request 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureInAlternate</a:Action>
    <a:MessageID>urn:uuid:6d43cb1b-cae1-4383-94bc-2cdfa4c183c0</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/Soap11</a:To>
  </s:Header>
  <s:Body>
    <SignatureIn xmlns="http://www.example.org/signature">Hello</SignatureIn>
  </s:Body>
```

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

### 3.2.1.3 Sample SENDER Request 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureInEmpty</a:Action
  >
    <a:MessageID>urn:uuid:6b6e4411-d156-4f21-a2e0-a6d5060f75ed</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/Soap11</a:To>
  </s:Header>
  <s:Body />
</s:Envelope>
```

### 3.2.1.4 Sample SENDER Request 4

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureInAlternateEmpty<
/a:Action>
    <a:MessageID>urn:uuid:752e028c-ad3c-4db7-8667-9b9e8f1ad9d2</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/Soap11</a:To>
  </s:Header>
  <s:Body />
</s:Envelope>
```

### 3.2.1.5 Sample SENDER Request 5

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:s="
http://schemas.xmlsoap.org/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
oap11</a:To>
  </s:Header>
  <s:Body />
</s:Envelope>
```

### 3.2.1.6 Sample SENDER Request 6

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureInMultipart</a:Action>
    <a:MessageID>urn:uuid:b81dfa10-4801-43e5-be6e-0b4304f9bef9</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/Soap11</a:To>
  </s:Header>
  <s:Body>
    <SignatureInRequest xmlns="http://www.example.org/signature">
      <SignatureInFirstPart
xmlns="http://www.example.org/signature">Hello</SignatureInFirstPart>
      <SignatureInSecondPart
xmlns="http://www.example.org/signature">World</SignatureInSecondPart>
    </SignatureInRequest>
  </s:Body>
</s:Envelope>
```

### 3.2.1.7 Sample SENDER Request 7

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderIn</a:Action>
    <h:SignatureInHeader
xmlns:h="http://www.example.org/signature">Hello</h:SignatureInHeader>
    <a:MessageID>urn:uuid:84991d60-6ec5-4add-a8a1-f4751f4d969b</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/Soap11</a:To>
  </s:Header>
  <s:Body>
    <SignatureInHeaderMember
xmlns="http://www.example.org/signature">World</SignatureInHeaderMember>
  </s:Body>
</s:Envelope>
```

### 3.2.1.8 Sample SENDER Request 8

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderIn</a:Action>
```

```

    <h:SignatureInHeader
xmlns:h="http://www.example.org/signature">Fault</h:SignatureInHeader>
    <a:MessageID>urn:uuid:3d4afe08-2de8-45e6-a803-f0f50ba98ac6</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.exanple.org/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11</a:To>
    </s:Header>
    <s:Body>
      <SignatureInHeaderMember
xmlns="http://www.example.org/signature">World</SignatureInHeaderMember>
    </s:Body>
</s:Envelope>

```

### 3.2.1.9 Sample RECEIVER Response 1

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureOut</a:Action>
    <a:RelatesTo>urn:uuid:80763f6e-999e-44bd-9d62-e7878eeleec6</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://www.example.org/signature">Sign1,
Parameter=Hello</SignatureOut>
  </s:Body>
</s:Envelope>

```

### 3.2.1.10 Sample RECEIVER Response 2

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureOutAlternate</a:A
ction>
    <a:RelatesTo>urn:uuid:6d43cb1b-cae1-4383-94bc-2cdfa4c183c0</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://www.example.org/signature">Sign2,
Parameter=Hello</SignatureOut>
  </s:Body>
</s:Envelope>

```

### 3.2.1.11 Sample RECEIVER Response 3

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureOutEmpty</a:Actio
n>
    <a:RelatesTo>urn:uuid:6b6e4411-d156-4f21-a2e0-a6d5060f75ed</a:RelatesTo>
  </s:Header>
  <s:Body>

```

```

    <SignatureOut xmlns="http://www.example.org/signature">Sign3
</SignatureOut>
  </s:Body>
</s:Envelope>

```

### 3.2.1.12 Sample RECEIVER Response 4

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureOutAlternateEmpty
</a:Action>
    <a:RelatesTo>urn:uuid:752e028c-ad3c-4db7-8667-9b9e8f1ad9d2</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://www.example.org/signature">Sign4
</SignatureOut>
  </s:Body>
</s:Envelope>

```

### 3.2.1.13 Sample RECEIVER Response 5

```

<s:Envelope xmlns:s=" http://schemas.xmlsoap.org/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>

```

### 3.2.1.14 Sample RECEIVER Response 6

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureOutMultipart</a:A
ction>
    <a:RelatesTo>urn:uuid:b81dfa10-4801-43e5-be6e-0b4304f9bef9</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOut xmlns="http://www.example.org/signature">Sign6,
Parameter=Hello, Parameter=World</SignatureOut>
  </s:Body>
</s:Envelope>

```

### 3.2.1.15 Sample RECEIVER Response 7

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderReply</a:Ac
tion>

```

```

    <h:SignatureOutHeader xmlns:h="http://www.example.org/signature">Sign7,
Parameter=Hello, Parameter=World</h:SignatureOutHeader>
    <a:RelatesTo>urn:uuid:84991d60-6ec5-4add-a8a1-f4751f4d969b</a:RelatesTo>
  </s:Header>
  <s:Body>
    <SignatureOutHeaderMember xmlns="http://www.example.org/signature">
Sign7, Parameter=Hello, Parameter=World</SignatureOutHeaderMember>
  </s:Body>
</s:Envelope>

```

### 3.2.1.16 Sample RECEIVER Response 8

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderFault</a:Ac
tion>
    <a:RelatesTo>urn:uuid:3d4afe08-2de8-45e6-a803-f0f50ba98ac6</a:RelatesTo>
  </s:Header>
  <s:Body>
    <s:Fault>
      <faultcode>s:Client</faultcode>
      <faultstring xml:lang="en-US">The creator of this fault did not specify
a Reason.</faultstring>
      <detail>
        <SignatureHeaderFaultContract
xmlns="http://www.example.org/signature"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <FaultAdditionalText>Fault: Fault, World</FaultAdditionalText>
        </SignatureHeaderFaultContract>
      </detail>
    </s:Fault>
  </s:Body>
</s:Envelope>

```

## 3.3 Test 1198

Test 1198 uses an rpc-literal binding with WS-Addressing 1.0. 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.

### 3.3.1 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.

### 3.3.1.1 Sample SENDER Request 1

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/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/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/Rpc</a:To>
  </s:Header>
  <s:Body>
    <Sign1 xmlns="http://www.example.org/signature" />
  </s:Body>
</s:Envelope>
```

### 3.3.1.2 Sample SENDER Request 2

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:s="
http://schemas.xmlsoap.org/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://example.org/Soap11/Rpc</a:To>
  </s:Header>
  <s:Body>
    <Sign2 xmlns=http://example.org />
  </s:Body>
</s:Envelope>
```

### 3.3.1.3 Sample SENDER Request 3

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/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/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/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>

```

#### 3.3.1.4 Sample SENDER Request 4

```

<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/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/WSAddressingCR_Service_WCF/WSAddr
essingSignature.svc/Soap11/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>

```

#### 3.3.1.5 Sample RECEIVER Response 1

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/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>
```

### 3.3.1.6 Sample RECEIVER Response 2

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/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>
```

### 3.3.1.7 Sample RECEIVER Response 3

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/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>
```

### 3.3.1.8 Sample RECEIVER Response 4

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://www.example.org/action/SignatureHeaderFaultRpc</a:
Action>
    <a:RelatesTo>urn:uuid:4c6d6ae8-8cb2-4870-b5b4-cfaecdf66bdb</a:RelatesTo>
  </s:Header>
  <s:Body>
    <s:Fault>
      <faultcode>s:Client</faultcode>
      <faultstring xml:lang="en-US">Fault Reason.</faultstring>
      <detail>
        <SignatureHeaderFaultContractRpc
xmlns="http://www.example.org/signature"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <FaultAdditionalTextRpc>Fault: Fault,
World</FaultAdditionalTextRpc>
        </SignatureHeaderFaultContractRpc>
      </detail>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

## 4 Acknowledgements

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

Mark Cowlshaw, Microsoft  
Asad Jawahar, Microsoft  
Vipul Modi, Microsoft  
Kirill Gavrylyuk, Microsoft  
Asir Vedamuthu, Microsoft  
Monica J. Martin, Microsoft

## 5 Revision History

Date	Revisions	Comments
27 August 2008	Add <a href="#">revisions</a> to reflect format for Basic Profile WG working draft.	Revisions integrated.
8 September 2008	Added Sample WSDL References and completed revisions consistent with those in BP 2 raised or found during the F2F in Sunnyvale 25-27 August 2008.	Revisions integrated.
9 September 2008	Added new test cases. Applied minor corrections found in review. Tests cover R2213, R2214, and R2744.	Revisions integrated.
24 October 2008	Remove MTOM tests	Revisions integrated.
3 November 2008	Changes to operation names for BaseDataTypes tests	Revisions integrated.
18 November 2008	Changes from BP F2F 11/11 – 11/13/2008	Revisions integrated.
24 November 2008	Approved as Working Group Draft.	Revisions integrated.