Web Services Addressing 1.0 - WSDL Binding

W3C Working Draft 16 February 2006

This version:
http://www.w3.org/TR/2006/WD-ws-addr-wsdl-20060216
Latest version:
http://www.w3.org/TR/ws-addr-wsdl
Previous versions:
http://www.w3.org/TR/2005/WD-ws-addr-wsdl-20050413
Editors:
Martin Gudgin, Microsoft Corp
Marc Hadley, Sun Microsystems, Inc
Tony Rogers, Computer Associates International, Inc
Ümit Yalçinalp, SAP AG

This document is also available in these non-normative formats: PDF, PostScript, XML, and plain text.

Copyright © 2006 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, trademark and document use rules apply.

Abstract

Web Services Addressing provides transport-neutral mechanisms to address Web services and messages. Web Services Addressing 1.0 - WSDL Binding (this document) defines how the abstract properties defined in Web Services Addressing 1.0 - Core are described using WSDL.

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This is the Last Call Working Draft of the Web Services Addressing 1.0 - WSDL Binding specification for review by W3C members and other interested parties. It has been produced by the Web Services Addressing Working Group (WG), which is part of the W3C Web Services Activity.

If the feedback is positive, the Working Group plans to submit this specification for consideration as a W3C Candidate Recommendation. Comments on this document are invited and are to be sent to the public public-ws-addressing-comments@w3.org mailing list (public archive). Comments can be sent until **31**

March 2006.

This documents addresses all the open issues against the previous Working Draft. In particular, the ability to describe asynchronous interactions has been incorporated (issue i059). A diff-marked version against the previous version of this document is available. For a detailed list of changes since the last publication of this document, please refer to appendix **C. Change Log** [p.26] . Issues about this document are documented in the Last Call issues list maintained by the Working Group.

Discussion of this document takes place on the public-ws-addressing@w3.org mailing list (public archive).

This document was produced under the 5 February 2004 W3C Patent Policy. The Working Group maintains a public list of patent disclosures made in connection with this document; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) with respect to this specification must disclose the information in accordance with section 6 of the W3C Patent Policy.

Publication as a Working Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

Table of Contents

1. Introduction [p.3]
   1.1 Notational Conventions [p.3]
   1.2 Namespaces [p.4]
2. Including WSDL Metadata in EPRs [p.5]
   2.1 Referencing WSDL Metadata from an EPR [p.5]
   2.2 Embedding WSDL Metadata in an EPR [p.6]
3. Indicating Use of WS-Addressing [p.7]
   3.1 UsingAddressing Extension Element [p.7]
      3.1.1 WSDL 2.0 Component Model Changes [p.9]
      3.1.2 Other Uses of UsingAddressing Extension Element [p.9]
   3.2 Anonymous Element [p.9]
      3.2.1 WSDL 2.0 Component Model Changes [p.11]
   3.3 WSDL SOAP Module [p.11]
4. Specifying Message Addressing Properties in WSDL [p.11]
   4.1 Destination [p.11]
   4.2 Action [p.12]
      4.2.1 Explicit Association [p.12]
      4.2.2 Default Action Pattern for WSDL 2.0 [p.13]
      4.2.3 WSDL 2.0 Component Model Changes [p.15]
      4.2.4 Default Action Pattern for WSDL 1.1 [p.15]
   4.3 Reference Parameters [p.18]
5. WS-Addressing and WSDL Message Exchange Patterns [p.18]
   5.1 WSDL 1.1 Message Exchange Patterns [p.18]

5.1.1 One-way [p.18]
5.1.2 Request-Response [p.19]
5.1.3 Notification [p.20]
5.1.4 Solicit-response [p.20]
5.2 WSDL 2.0 Message Exchange Patterns [p.20]
5.2.1 In-only [p.20]
5.2.2 Robust In-only [p.20]
5.2.3 In-out [p.22]
5.2.4 In-optional-out [p.23]
5.2.5 Out-only [p.23]
5.2.6 Robust Out-only [p.23]
5.2.7 Out-in [p.23]
5.2.8 Out-optional-in [p.23]
6. References [p.23]

Appendices

A. Acknowledgements [p.24] (Non-Normative)
B. Compatibility of [action] with previous versions of WS-Addressing [p.25] (Non-Normative)
C. Change Log [p.26] (Non-Normative)
C.1 Changes Since Third Working Draft [p.26]
C.2 Changes Since Second Working Draft [p.27]
C.3 Changes Since First Working Draft [p.28]
C.4 Changes Since Submission [p.28]

---

1. Introduction

Web Services Addressing 1.0 - Core[*WS-Addressing-Core [p.23]*] defines a set of abstract properties and an XML Infoset [*XML Information Set [p.24]*] representation thereof to reference Web service endpoints and to facilitate end-to-end addressing of endpoints in messages. Web Services Addressing 1.0 - WSDL Binding (this document) defines how the abstract properties defined in Web Services Addressing 1.0 - Core are described using WSDL. WS-Addressing is designed to be able to work with WSDL 2.0 [*WSDL 2.0 [p.23]*] and also (for backwards compatibility with WSDL 1.1 [*WSDL 1.1 [p.24]*] described services.

1.1 Notational Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [*IETF RFC 2119 [p.23]*].

When describing abstract data models, this specification uses the notational convention used by the XML Infoset [*XML Information Set [p.24]*]. Specifically, abstract property names always appear in square brackets (e.g., [some property]).

When describing concrete XML schemas [*XML Schema Structures [p.24]* , *XML Schema Datatypes [p.24]*], this specification uses the notational convention of WS-Security [*WS-Security [p.24]*]. Specifically, each member of an element's [children] or [attributes] property is described using an XPath-like notation (e.g., /x:MyHeader/x:SomeProperty/@value1). The use of {any} indicates the presence of an element wildcard (<xs:any/>). The use of @{any} indicates the presence of an attribute wildcard (<xs:anyAttribute/>).

1.2 Namespaces

This specification uses a number of namespace prefixes throughout; they are listed in Table 1-1 [p.4] . Note that the choice of any namespace prefix is arbitrary and not semantically significant (see [*XML Namespaces [p.24]*]).

Table 1-1. Prefixes and Namespaces used in this specification

Prefix	**Namespace**
S	http://www.w3.org/2003/05/soap-envelope
S11	http://schemas.xmlsoap.org/soap/envelope
wsa	http://www.w3.org/2005/08/addressing
wsaw	http://www.w3.org/2006/02/addressing/wsdl
xs	http://www.w3.org/2001/XMLSchema
wsdl	Either http://www.w3.org/2006/01/wsdl or http://schemas.xmlsoap.org/wsdl/ depending on context
wsdl20	http://www.w3.org/2006/01/wsdl
wsdl11	http://schemas.xmlsoap.org/wsdl/
soap11	http://schemas.xmlsoap.org/wsdl/soap/

The working group intends to update the value of the Web Services Addressing 1.0 - WSDL Binding namespace URI each time a new version of this document is published until such time that the document reaches Candidate Recommendation status. Once it has reached Candidate Recommendation status, the working group intends to maintain the value of the Web Services Addressing 1.0 - WSDL Binding namespace URI that was assigned in the Candidate Recommendation unless significant changes are made that impact the implementation of the specification.

WS-Addressing is defined in terms of the XML Information Set [*XML Information Set [p.24]*]. WS-Addressing may be used with SOAP [*SOAP 1.2 Part 1: Messaging Framework [p.24]*] as described in Web Services Addressing 1.0 - SOAP Binding[*WS-Addressing-SOAP [p.23]*]. The examples in this specification use an XML 1.0 [*XML 1.0 [p.24]*] representation but this is not a requirement.

All information items defined by this specification are identified by the XML namespace URI [*XML Namespaces [p.24]*] "http://www.w3.org/2006/02/addressing/wsdl". A normative XML Schema [*XML Schema Structures [p.24]* , *XML Schema Datatypes [p.24]*] document can be obtained by dereferencing the XML namespace URI.

2. Including WSDL Metadata in EPRs

An EPR's metadata section can contain a reference to WSDL metadata, can include embedded WSDL metadata, or both.

2.1 Referencing WSDL Metadata from an EPR

The WSDL binding of Web Services Addressing introduces the following element and attribute information items for referencing WSDL metadata from an EPR's metadata section:

wsaw:InterfaceName

> A QName identifying a description of the sequences of messages that a service sends and/or receives. This corresponds to a WSDL 2.0 interface or, for backwards compatibility, a WSDL 1.1 port type. When this element is included in an EPR, the EPR is considered to be specific to the interface or port type it identifies.

wsaw:ServiceName

> A QName that identifies the set of endpoints at which a particular Web service is deployed. The set of endpoints is represented by a service in WSDL 2.0 or, for backwards compatibility, a WSDL 1.1 service.

wsaw:ServiceName/@EndpointName

> An NCName that identifies one endpoint amongst the set identified by by the service name above. An endpoint is represented by an endpoint in WSDL 2.0 or, for backwards compatibility, a port in WSDL 1.1. When this attribute is specified, the EPR is considered to be specific to the endpoint or port it identifies.

The element information items defined above are used in an EPR's metadata section. The following shows an example endpoint reference. This references the interface named "fabrikam:Inventory" at the endpoint IRI "http://example.com/www.fabrikam/acct". Note the use of the WSDL[*WSDL 2.0 [p.23]*] wsdlLocation attribute.

Example 2-1. Example endpoint reference.

```
<wsa:EndpointReference
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:fabrikam="http://example.com/fabrikam">
  <wsa:Address>http://example.com/fabrikam/acct</wsa:Address>
  <wsa:Metadata
      xmlns:wsdli="http://www.w3.org/2006/01/wsdl-instance"
      wsdli:wsdlLocation="http://example.com/fabrikam http://example.com/fabrikam.wsdl">
    <wsaw:InterfaceName>fabrikam:Inventory</wsaw:InterfaceName>
  </wsa:Metadata>
</wsa:EndpointReference>
```

2.2 Embedding WSDL Metadata in an EPR

WSDL 2.0 or, for backwards compatibility, 1.1 definitions can be embedded in the metadata section of an EPR to provide a consuming application with WSDL information that applies to the referenced endpoint. To do so, the creator of an EPR MAY include a WSDL 2.0 description element (or a WSDL 1.1 definitions element) in the metadata property of the EPR. The semantics of the embedded WSDL is as defined by the WSDL 2.0 or 1.1 specifications.

In particular, embedding a WSDL service component description MAY be used by EPR issuers to indicate the presence of alternative addresses and protocol bindings to access the referenced endpoint. The alternatives are provided by the different endpoints of the embedded service. In the case of WSDL 1.1, additional ports may be conveyed by the WSDL 1.1 service definition which are not alternative access channels to the endpoint. In that case, if the InterfaceName or ServiceName elements are also included in the metadata section of the EPR, only the ports with the same interface as that specified are to be considered alternative access channels.

If the ServiceName element appears in the EPR's [metadata] and an embedded WSDL service component is also provided inside a descriptions or definitions component, then the ServiceName SHOULD match the name of (one or more of) the WSDL service(s) included therein; the endpoint (port) name SHOULD match as well if present. The behavior of an EPR consumer when the ServiceName doesn't match an embedded description is undefined.

Example 2-2. An EPR containing WSDL 2.0 metadata

```
<wsa:EndpointReference
    xmlns:wsa="http://www.w3.org/2005/08/addressing">

  <wsa:Address>http://example.com/fabrikam/acct</wsa:Address>
  <wsa:Metadata
      xmlns:wsdli="http://www.w3.org/2006/01/wsdl-instance"
      wsdli:wsdlLocation="http://example.com/fabrikam http://example.com/fabrikam.wsdl">
    <wsdl20:description
        targetNamespace="http://example.com/fabrikam"
        xmlns:fabrikam="http://example.com/fabrikam"
        xmlns:abc="http://www.abccorp.com/"
        xmlns:wsdl20="http://www.w3.org/2006/01/wsdl">
      <wsdl20:import namespace="http://example.com/fabrikam/"
          location="http://example.com/fabrikam/fabrikam.wsdl/"/>
      <wsdl20:import namespace="http://www.abccorp.com/"
          location="http://www.abccorp.com/abc.wsdl"/>
      <wsdl20:service name="InventoryService"
```

```
          interface="fabrikam:Inventory">
        <wsdl20:endpoint name="ep1" binding="abc:soap-http-binding"
            address="http://example.com/fabrikam/acct"/>
        <wsdl20:endpoint name="ep2" binding="abc:iiop"
            address="..."/>
        <wsdl20:endpoint name="ep3"
            binding="abc:soap-http-binding"
            address="http://alt.example.com/fabrikam/acct"/>
      </wsdl20:service>
    </wsdl20:description>
  </wsa:Metadata>
</wsa:EndpointReference>
```

Example 2-3. An EPR containing WSDL 1.1 metadata

```
<wsa:EndpointReference
    xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <wsa:Address>http://example.com/fabrikam/acct</wsa:Address>
  <wsa:Metadata>
    <wsdl11:definitions targetNamespace="http://example.com/fabrikam"
        xmlns:fabrikam="http://example.com/fabrikam"
        xmlns:abc="http://www.abccorp.com/"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns:iiop="http://www.iiop.org/"
        xmlns:wsdl11="http://schemas.xmlsoap.org/wsdl/">
      <wsdl11:import namespace="http://example.com/fabrikam"
          location="http://example.com/fabrikam/fabrikam.wsdl"/>
      <wsdl11:import namespace="http://www.abccorp.com/"
          location="http://www.abccorp.com/abc.wsdl"/>
      <wsdl11:service name="InventoryService">
        <wsdl11:port name="ep1" binding="abc:soap-http-binding">
          <soap:address location="http://example.com/fabrikam/acct"/>
        </wsdl11:port>
        <wsdl11:port name="ep2" binding="abc:iiop">
          <iiop:address location="..."/>
        </wsdl11:port>
      </wsdl11:service>
    </wsdl11:definitions>
  </wsa:Metadata>
</wsd:EndpointReference>
```

3. Indicating Use of WS-Addressing

This specification supports two mechanisms for indicating, in a WSDL description, that the endpoint conforms to the WS-Addressing specification.

3.1 UsingAddressing Extension Element

WS-Addressing defines an empty global element, wsaw:UsingAddressing, that may be used to indicate that an endpoint conforms to the WS-Addressing specification. The wsdl:required attribute MAY be used to indicate whether WS-Addressing Message Addressing Properties are required in messages received from service requesters. Table 3-1 [p.8] outlines the requirements on messages sent from an endpoint based on the contents of any preceding input message and how the use of addressing is indicated in the

WSDL.

Table 3-1. MAPs Present in output message

MAPs in Input message	UsingAddressing Present		UsingAddressing Not Present
	wsdl:required="true"	wsdl:required="false"	
Yes, using SOAP headers with a soap:mustUnderstand value of "true"	REQUIRED	REQUIRED	REQUIRED or fault
Yes, using another protocol or using SOAP headers with a soap:mustUnderstand value of "false"	REQUIRED	REQUIRED	OPTIONAL
No	Fault	OPTIONAL. If using SOAP, MAP headers MUST NOT have a soap:mustUnderstand attribute with a value of "true"	OPTIONAL. If using SOAP, MAP headers MUST NOT have a soap:mustUnderstand attribute with a value of "true"

If WS-A is engaged, use of the message addressing properties MUST be fully compliant with this specification; in particular, senders MUST use all message addressing properties mandated by the Web Services Addressing 1.0 - Core[*WS-Addressing-Core [p.23]*], applicable WS-Addressing protocol bindings (e.g. Web Services Addressing 1.0 - SOAP Binding[*WS-Addressing-SOAP [p.23]*]), and this specification, and MUST follow all applicable WS-Addressing normative requirements.

The wsaw:UsingAddressing element SHOULD appear as a child of the wsdl:binding element. Alternatively, the wsaw:UsingAddressing element MAY instead be included as a child of the wsdl20:endpoint (or wsdl11:port) when an endpoint intends to indicate compliance with WS-Addressing for a specific endpoint only.

The inclusion of the wsaw:UsingAddressing element indicates that the applicable WS-Addressing specifications are supported and allows use of anonymous or non-anonymous URIs as addresses in an EPR. Specifically, when included in a SOAP binding, the wsaw:UsingAddressing marker identifies the use of Web Services Addressing 1.0 bound to SOAP as defined by Web Services Addressing 1.0 - SOAP Binding[*WS-Addressing-SOAP [p.23]*].The presence of this element may extend the semantics of the endpoint’s WSDL binding.

Example 3-1. Indicating use of WS-Addressing using wsaw:UsingAddressing in WSDL 2.0

```
<binding name="reservationSOAPBinding"
    interface="tns:reservationInterface"
    type="http://www.w3.org/2006/01/wsdl/soap"
    wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP">
  <wsaw:UsingAddressing wsdl:required="true" />
  <operation ref="tns:opCheckAvailability"
      wsoap:mep="http://www.w3.org/2003/05/soap/mep/request-response" />
  <fault ref="tns:invalidDataFault" wsoap:code="soap:Sender" />
</binding>
```

Example 3-2. Indicating use of WS-Addressing using wsaw:UsingAddressing in WSDL 1.1

```
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <wsaw:UsingAddressing wsdl:required="true" />
  <operation name="GetLastTradePrice">
    <soap:operation soapaction="http://example.com/GetLastTradePrice" />
    <input>
      <soap:body use="literal" />
    </input>
    <output>
      <soap:body use="literal" />
    </output>
  </operation>
</binding>
```

3.1.1 WSDL 2.0 Component Model Changes

Use of WS-Addressing adds the following REQUIRED properties to the WSDL 2.0 component model:

- A property of the binding or endpoint named {addressing required} of type xs:boolean. The property value is the value of the wsdl:required attribute information item on the wsaw:UsingAddressing extension element, if present; otherwise "false".

3.1.2 Other Uses of UsingAddressing Extension Element

The wsaw:UsingAddressing element may also be used in other contexts (e.g., as a policy assertion in a policy framework). Its use and that of related elements and attributes including wsaw:Anonymous (see **3.2 Anonymous Element** [p.9]) and wsaw:Action (see **4.2.1 Explicit Association** [p.12]) in such contexts is semantically equivalent to the use of wsaw:UsingAddressing as a WSDL extension.

Note that the association of wsaw:UsingAddressing to WSDL constructs where the wsaw:UsingAddressing WSDL extension element is not allowed is not meaningful.

3.2 Anonymous Element

WS-Addressing defines a wsaw:Anonymous element that may only be used in conjunction with wsaw:UsingAddressing (or its equivalent wsoap:module) element. The usage of wsaw:Anonymous element is associated with the usage constraints specified for the wsaw:UsingAddressing element. Hence, it MUST NOT contain the wsdl:required attribute.

The inclusion of wsaw:Anonymous without inclusion of wsaw:UsingAddressing has no normative meaning and is only informational. In other words, the inclusion of wsaw:Anonymous element in WSDL alone does not imply a requirement on clients to use Message Addressing Properties in messages it sends to the service. A client, however, MAY include Message Addressing Properties in the messages it sends, either on its own initiative or as described by other elements of the service contract, regardless of the presence or absence of a wsaw:UsingAddressing element.

The wsaw:Anonymous element MAY have three distinct values that indicate three different levels of support for handling anonymous addresses in EPRs. In the following text, the term response endpoint EPR refers to the [reply endpoint] and [fault endpoint] message addressing properties collectively.

- "optional": This value indicates that a response endpoint EPR in a request message MAY contain an anonymous URI as an address.

  This is the default value, omission of a wsaw:Anonymous element is equivalent to inclusion with this value.

- "required":This value indicates that all response endpoint EPRs in a request message MUST always use anonymous URI as an address.

  If a response endpoint EPR does not contain the anonymous URI as an address value, then a predefined InvalidAddressingHeader fault defined in Web Services Addressing 1.0 - SOAP Binding [*WS-Addressing-SOAP [p.23]*] MUST be generated.

- "prohibited":This value indicates that any response EPRs in a request message MUST NOT use anonymous URI as an address.

  If a response endpoint EPR contains the anonymous URI as an address value, then a predefined InvalidAddressingHeader fault defined in Web Services Addressing 1.0 - SOAP Binding [*WS-Addressing-SOAP [p.23]*] MUST be generated.

This element MAY appear as a child of an operation element in a binding element in WSDL 1.1, or as a binding operation extension element in WSDL 2.0.

Example 3-3. Indicating use of anonymous addresses using wsaw:Anonymous.

```
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <wsaw:UsingAddressing wsdl:required="true" />
  <operation name="GetLastTradePrice">
    <soap:operation soapaction="http://example.com/GetLastTradePrice" />
    <input>
      <soap:body use="literal" />
    </input>
    <output>
      <soap:body use="literal" />
    </output>
    <wsaw:Anonymous>required</wsaw:Anonymous>
  </operation>
  <operation name="someOther">...</operation>
</binding>
```

3.2.1 WSDL 2.0 Component Model Changes

Use of the Anonymous element adds the following REQUIRED properties to the WSDL 2.0 component model:

- A property of the binding operation component {anonymous required} of type xs:string. The property value is the value of the wsaw:Anonymous element information item, if present; otherwise "optional".

3.3 WSDL SOAP Module

In WSDL 2.0, the wsoap:module construct may be used to declare the use of the WS-Addressing 1.0 Module for the SOAP binding. The meaning of such a wsoap:module declaration is semantically equivalent to wsaw:UsingAddressing in this case. Note that this module is not meaningful when used on WSDL constructs where wsaw:UsingAddressing is not allowed.

The WS-Addressing 1.0 SOAP Module is described in Web Services Addressing 1.0 - SOAP Binding[*WS-Addressing-SOAP [p.23]*] and is identified with the following URI: http://www.w3.org/2005/08/addressing/module

Example 3-4. Indicating use of WS-Addressing using wsoap:module in WSDL 2.0

```
<binding name="reservationSOAPBinding"
    interface="tns:reservationInterface"
    type="http://www.w3.org/2006/01/wsdl/soap"
    wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP">
  <wsoap:module uri="http://www.w3.org/2005/08/addressing/module" required="true" />
  <operation ref="tns:opCheckAvailability"
      wsoap:mep="http://www.w3.org/2003/05/soap/mep/request-response">
    <wsaw:Anonymous>required</wsaw:Anonymous>
  </operation>
  <fault ref="tns:invalidDataFault" wsoap:code="soap:Sender" />
</binding>
```

4. Specifying Message Addressing Properties in WSDL

This section describes how the values of certain message addressing properties may be specified in WSDL. In some cases the values of message addressing properties are specified using existing WSDL constructs, in other cases new WSDL extensions are defined for that purpose.

4.1 Destination

In the absence of additional runtime information, the value of the [destination] message addressing property for a message sent to an endpoint MUST match the value of the {address} property of the endpoint component (WSDL 2.0) or the address value provided by the relevant port extension (WSDL 1.1). For a SOAP 1.1 port described using WSDL 1.1, the value is provided by the location attribute of the soap11:address extension element.

Additional runtime information could override the value of the [destination] message addressing property for messages sent to an endpoint, e.g. a runtime exchange might result in a redirection to a different EPR. Note that WS-Addressing does not define any normative mechanism for such redirection.

4.2 Action

WS-Addressing defines two mechanisms to associate a value of the [action] property with input, output and fault elements within a WSDL description: explicit and defaulting. Explicit association is described in section **4.2.1 Explicit Association** [p.12] ; action defaulting (where a unique value for the [action] property is automatically generated) is described in section **4.2.4 Default Action Pattern for WSDL 1.1** [p.15] for WSDL 1.1 and section **4.2.2 Default Action Pattern for WSDL 2.0** [p.13] for WSDL 2.0.

Ensuring that there is sufficient information within a message to distinguish which WSDL operation it is associated with is specified as a best practice in WSDL 2.0*WSDL 2.0 [p.23]* . The [action] property provides a mechanism to fulfill that best practice.

4.2.1 Explicit Association

WS-Addressing defines a global attribute, wsaw:Action, that may be used to explicitly define the value of the [action] property for messages in a WSDL description. The type of the attribute is xs:anyURI and it is used as an extension on the WSDL input, output and fault elements. A SOAP binding can specify SOAPAction values for the input messages of operations. In the absence of a wsaw:Action attribute on a WSDL input element where a SOAPAction value is specified, the value of the [action] property for the input message is the value of the SOAPAction specified. Web Services Addressing 1.0 - SOAP Binding[*WS-Addressing-SOAP [p.23]*] specifies restrictions on the relationship between the values of [action] and SOAPAction for SOAP 1.1 and SOAP 1.2.

The inclusion of wsaw:Action without inclusion of wsaw:UsingAddressing has no normative intent and is only informational. In other words, the inclusion of wsaw:Action attributes in WSDL alone does not imply a requirement on clients to use Message Addressing Properties in messages it sends to the service. A client, however, MAY include Message Addressing Properties in the messages it sends, either on its own initiative or as described by other elements of the service contract, regardless of the presence or absence of wsaw:UsingAddressing. Other specifications defining the value of [action] are under no constraint to be consistent with wsaw:Action.

For example consider the following WSDL excerpt:

Example 4-1. Explicit specification of wsaw:Action value in a WSDL 2.0 description.

```
<definitions targetNamespace="http://example.com/stockquote" ...>
  ...
  <interface name="StockQuoteInterface">
    <operation name="GetLastTradePrice" pattern="http://www.w3.org/2006/01/wsdl/in-out">
      <input element="tns:GetTradePricesInput" messageLabel="In"
            wsaw:Action="http://example.com/GetQuote"/>
      <output element="tns:GetTradePricesOutput" messageLabel="Out"
            wsaw:Action="http://example.com/Quote"/>
```

```
    </operation>
  </interface>
  ...
</definitions>
```

The action for the input of the GetLastTradePrice operation within the StockQuoteInterface is explicitly defined to be http://example.com/GetQuote. The action for the output of this same operation is http://example.com/Quote.

Example 4-2. Explicit specification of wsa:Action value in a WSDL 1.1 description.

```
<definitions targetNamespace="http://example.com/stockquote" ...>
  ...
  <portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
      <input message="tns:GetTradePricesInput"
            wsaw:Action="http://example.com/GetQuote"/>
      <output message="tns:GetTradePricesOutput"
            wsaw:Action="http://example.com/Quote"/>
    </operation>
  </portType>
  ...
</definitions>
```

The action for the input of the GetLastTradePrice operation within the StockQuotePortType is explicitly defined to be http://example.com/GetQuote. The action for the output of this same operation is http://example.com/Quote.

4.2.2 Default Action Pattern for WSDL 2.0

In the absence of the wsa:Action attribute, the following pattern is used in WSDL 2.0 documents to construct a default action for inputs and outputs. The general form of an action URI is as follows:

Example 4-3. Structure of defaulted wsa:Action IRI in WSDL 2.0.

```
[target namespace][delimiter][interface name][delimiter][operation name][direction token]
```

For fault messages, the general form of an action IRI is as follows:

Example 4-4. Structure of default wsa:Action IRI for faults

```
[target namespace][delimiter][interface name][delimiter][fault name]
```

Where:

[delimiter]

> is ":" when the [target namespace] is a URN, otherwise "/". Note that for IRI schemes other than URNs which aren't path-based (i.e. those that outlaw the "/" character), the default action value may not conform to the rules of the IRI scheme. Authors are advised to specify explicit values in the WSDL in this case.

[target namespace]

is the {target namespace} of the interface. If [target namespace] ends with a "/" an additional "/" is not added.

[interface name]

is the {name} of the interface.

[operation name]

is the {name} of the operation.

[fault name]

is the {name} of the fault.

[direction token]

- Empty ("") where the operation's {message exchange pattern} is "http://www.w3.org/2006/01/wsdl/in-only", "http://www.w3.org/2006/01/wsdl/robust-in-only", "http://www.w3.org/2006/01/wsdl/out-only", or "http://www.w3.org/2006/01/wsdl/robust-out-only".
- "Request" where the operation's {message exchange pattern} is "http://www.w3.org/2006/01/wsdl/in-out" or "http://www.w3.org/2006/01/wsdl/in-opt-out" and the message reference's {message label} = 'in'.
- "Solicit" where the operation's {message exchange pattern} is "http://www.w3.org/2006/01/wsdl/out-in" or "http://www.w3.org/2006/01/wsdl/out-opt-in" and the message reference's {message label} = 'out'.
- "Response" where the operation's {message exchange pattern} is "http://www.w3.org/2006/01/wsdl/in-out" or "http://www.w3.org/2006/01/wsdl/in-opt-out" and the message reference's {message label} = 'out'.
- "Response" where the operation's {message exchange pattern} is "http://www.w3.org/2006/01/wsdl/out-in", or "http://www.w3.org/2006/01/wsdl/out-opt-in" and the message reference's {message label} = 'in'.
- {message label} where the {message exchange pattern} is not one of the MEP IRIs defined in WSDL 2.0 Part 2.

For example consider the following WSDL excerpt:

Example 4-5. Example WSDL without explicit wsa:Action values with explicit message names.

```
<definitions targetNamespace="http://example.com/stockquote" ...>
  ...
  <interface name="StockQuoteInterface">
    <operation name="GetLastTradePrice" pattern="http://www.w3.org/2006/01/wsdl/in-out">
      <input element="tns:GetTradePricesInput" messageLabel="in" name="GetQuote"/>
      <output element="tns:GetTradePricesOutput" messageLabel="out" name="Quote"/>
    </operation>
  </portType>
  ...
</definitions>
```

[targetNamespace] = http://example.com/stockquote

[interface name] = StockQuoteInterface

[operation name] = GetLastTradePrice

[direction token] for input is Request

[direction token] for output is Response

Applying the pattern above with these values we have:

input action = http://example.com/stockquote/StockQuoteInterface/GetLastTradePriceRequest

output action = http://example.com/stockquote/StockQuoteInterface/GetLastTradePriceResponse

4.2.3 WSDL 2.0 Component Model Changes

Use of WS-Addressing adds the following REQUIRED properties to the WSDL 2.0 component model:

- A property of the Interface Message Reference and Interface Fault components named {action}. The property is of type xs:anyURI. The property value is the value of the wsaw:action attribute information item, if present; otherwise the default value computed following the rules from section **4.2.2 Default Action Pattern for WSDL 2.0** [p.13] .

4.2.4 Default Action Pattern for WSDL 1.1

A default pattern is also defined for backwards compatibility with WSDL 1.1. In the absence of the wsa:Action attribute, the following pattern is used to construct a default action for inputs and outputs. The general form of an action IRI is as follows:

Example 4-6. Structure of defaulted wsa:Action IRI.

```
[target namespace][delimiter][port type name][delimiter][input|output name]
```

For fault messages, the general form of an action IRI is as follows:

Example 4-7. Structure of default wsa:Action IRI for faults

```
[target namespace][delimiter][port type name][delimiter][operation name][delimiter]Fault[delimiter][fault name]
```

Where:

[delimiter]

is ":" when the [target namespace] is a URN, otherwise "/". Note that for IRI schemes other than URNs which aren't path-based (i.e. those that outlaw the "/" character), the default action value may not conform to the rules of the IRI scheme. Authors are advised to specify explicit values in the WSDL in this case.

"Fault"

is a literal character string to be included in the action.

[target namespace]

is the target namespace (/definition/@targetNamespace). If [target namespace] ends with a "/" an additional "/" is not added.

[port type name]

is the name of the port type (/definition/portType/@name).

[input|output name]

is the name of the element as defined in Section 2.4.5 of WSDL 1.1.

[fault name]

is the name of the fault (/definition/porttype/operation/fault/@name).

For example consider the following WSDL excerpt:

Example 4-8. Example WSDL without explicit wsa:Action values with explicit message names.

```
<definitions targetNamespace="http://example.com/stockquote" ...>
  ...
  <portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
      <input message="tns:GetTradePricesInput" name="GetQuote"/>
      <output message="tns:GetTradePricesOutput" name="Quote"/>
          <fault message="tns:ErrorMessage" name="Error"/>
    </operation>
  </portType>
  ...
</definitions>
```

[targetNamespace] = http://example.com/stockquote

[port type name] = StockQuotePortType

[input name] = GetQuote

[output name] = Quote

[fault name] = Error

Applying the pattern above with these values we have:

input action = http://example.com/stockquote/StockQuotePortType/GetQuote

output action = http://example.com/stockquote/StockQuotePortType/Quote

fault action = http://example.com/stockquote/StockQuotePortType/GetLastTradePriceFault/Error

WSDL defines rules for a default input or output name if the name attribute is not present. Consider the following example:

Example 4-9. Example WSDL without explicit wsa:Action values or explicit message names.

```
<definitions targetNamespace="http://example.com/stockquote" ...>
  ...
  <portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
      <input message="tns:GetTradePricesInput"/>
      <output message="tns:GetTradePricesOutput"/>
    </operation>
  </portType>
  ...
</definitions>
```

[targetNamespace] = http://example.com/stockquote

[port type name] = StockQuotePortType

According to the rules defined in Section 2.4.5 of WSDL 1.1, if the name attribute is absent for the input of a request response operation the default value is the name of the operation with "Request" appended.

[input name] = GetLastTradePriceRequest

Likewise, the output defaults to the operation name with "Response" appended.

[output name] = GetLastTradePriceResponse

Applying the pattern above with these values we have:

input action = http://example.com/stockquote/StockQuotePortType/GetLastTradePriceRequest

output action = http://example.com/stockquote/StockQuotePortType/GetLastTradePriceResponse

4.3 Reference Parameters

The wsa:ReferenceParameters element (see Web Services Addressing 1.0 - Core[*WS-Addressing-Core [p.23]*]) MAY be used as an extension child element of the wsdl20:endpoint or wsdl11:port elements. When present, the value of the [reference parameters] message addressing property for a message sent to an endpoint MUST include the contents of the wsa:ReferenceParameters WSDL extension element.

5. WS-Addressing and WSDL Message Exchange Patterns

This section describes which of the core message properties are mandatory or optional for messages in the various MEPs defined by WSDL.

5.1 WSDL 1.1 Message Exchange Patterns

For backwards compatibility, this section describes which of the core message properties are mandatory or optional for messages in the various MEPs defined by WSDL 1.1.

5.1.1 One-way

This is a straightforward one-way message. No responses are expected but related messages could be sent as part of other message exchanges.

Table 5-1. Message addressing properties for one way message.

Property	**Mandatory**	**Description**
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[message id]	N	Unique identifier for this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[relationship]	N	Indicates relationship to a prior message. Unused in this MEP, but may be included to facilitate longer running message exchanges.

5.1.2 Request-Response

This is request-response. A reply is expected hence mandating [reply endpoint] in the request message. The response message might be a fault.

Table 5-2. Message addressing properties for request message.

Property	Mandatory	Description
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	Y	Intended receiver for the reply to this message.
[fault endpoint]	N	Intended receiver for faults related to this message. May be included to direct fault messages to a different endpoint than [reply endpoint].
[message id]	Y	Unique identifier for this message. Used in the [relationship] property of the reply message.
[relationship]	N	Indicates relationship to a prior message. Unused in this MEP, but may be included to facilitate longer running message exchanges.

Table 5-3. Message addressing properties for response message.

Property	Mandatory	Description
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[message id]	N	Unique identifier for this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[relationship]	Y	Indicates that this message is a reply to the request message using the request message [message id] value and the predefined `http://www.w3.org/2005/08/addressing/reply` IRI.

5.1.3 Notification

From the WS-Addressing perspective this MEP is the same as One-way. The properties defined in **5.1.1 One-way** [p.18] apply to this MEP also.

5.1.4 Solicit-response

From the WS-Addressing perspective this MEP is the same as Request-response. The properties defined in **5.1.2 Request-Response** [p.19] apply to this MEP also.

5.2 WSDL 2.0 Message Exchange Patterns

This section describes which of the core message properties are mandatory or optional for messages in the various MEPs defined by WSDL 2.0 [*WSDL 2.0 Adjuncts [p.23]*].

5.2.1 In-only

This is a straightforward one-way message. No responses are expected but related messages could be sent as part of other message exchanges.

Table 5-4. Message addressing properties for in message.

Property	**Mandatory**	**Description**
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[message id]	N	Unique identifier for this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[relationship]	N	Indicates relationship to a prior message. Unused in this MEP, but may be included to facilitate longer running message exchanges.

5.2.2 Robust In-only

This one-way MEP allows fault messages. The [message id] property is needed in the initial message in order to be able to correlate any fault with that message.

Table 5-5. Message addressing properties for in message.

Property	**Mandatory**	**Description**
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message.
[message id]	Y	Unique identifier for this message. Used in the [relationship] property of any resulting fault message.
[relationship]	N	Indicates relationship to a prior message. Unused in this MEP, but may be included to facilitate longer running message exchanges.

Table 5-6. Message addressing properties for fault message.

Property	**Mandatory**	**Description**
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[message id]	N	Unique identifier for this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[relationship]	Y	Indicates that this message is a response to the in message using the in message [message id] value and the predefined `http://www.w3.org/2005/08/addressing/reply` IRI.

5.2.3 In-out

This is a two-way MEP. A reply is expected hence mandating [reply endpoint] in the request message. The response message might be a fault.

Table 5-7. Message addressing properties for in message.

Property	Mandatory	Description
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	Y	Intended receiver for the reply to this message.
[fault endpoint]	N	Intended receiver for faults related to this message. May be included to direct fault messages to a different endpoint than [reply endpoint].
[message id]	Y	Unique identifier for this message. Used in the [relationship] property of the out message.
[relationship]	N	Indicates relationship to a prior message. Unused in this MEP, but may be included to facilitate longer running message exchanges.

Table 5-8. Message addressing properties for out message.

Property	Mandatory	Description
[destination]	Y	Provides the address of the intended receiver of this message
[action]	Y	Identifies the semantics implied by this message
[source endpoint]	N	Message origin. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[message id]	N	Unique identifier for this message. Unused in this MEP, but may be included to facilitate longer running message exchanges.
[relationship]	Y	Indicates that this message is a response to the in message using the in message [message id] value and the predefined `http://www.w3.org/2005/08/addressing/reply` IRI.

5.2.4 In-optional-out

This MEP differs from the In-out MEP in that the subsequent message is optional. This difference doesn't affect the message properties so the properties defined in **5.2.3 In-out** [p.22] apply to this MEP also

5.2.5 Out-only

From the WS-Addressing perspective this MEP is the same as In-only. The properties defined in **5.2.1 In-only** [p.20] apply to this MEP also.

5.2.6 Robust Out-only

From the WS-Addressing perspective this MEP is the same as Robust In-only. The properties defined in **5.2.2 Robust In-only** [p.20] apply to this MEP also.

5.2.7 Out-in

From the WS-Addressing perspective this MEP is the same as In-out. The properties defined in **5.2.3 In-out** [p.22] apply to this MEP also.

5.2.8 Out-optional-in

This MEP differs from the Out-in MEP in that the subsequent message is optional. This difference doesn't affect the message properties so the properties defined in **5.2.3 In-out** [p.22] apply to this MEP also

6. References

[WS-Addressing-Core]
Web Services Addressing 1.0 - Core, M. Gudgin, M. Hadley, Editors.

[WS-Addressing-SOAP]
Web Services Addressing 1.0 - SOAP Binding, M. Gudgin, M. Hadley, Editors.

[WSDL 2.0]
Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, R. Chinnici, J. J. Moreau, A. Ryman, S. Weerawarana, Editors. World Wide Web Consortium, 6 January 2006. This version of the WSDL 2.0 specification is http://www.w3.org/TR/2006/CR-wsdl20-20060106/. The latest version of WSDL 2.0 is available at http://www.w3.org/TR/wsdl20.

[WSDL 2.0 Adjuncts]
Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts, R. Chinnici, H. Haas, A. Lewis, J. J. Moreau, D. Orchard, S. Weerawarana, Editors. World Wide Web Consortium, 6 January 2006. This version of the WSDL 2.0 Adjuncts specification is http://www.w3.org/TR/2006/CR-wsdl20-adjuncts-20060106/. The latest version of WSDL 2.0 Adjuncts is available at http://www.w3.org/TR/wsdl20-adjuncts.

[IETF RFC 2119]
Key words for use in RFCs to Indicate Requirement Levels, S. Bradner, Author. Internet Engineering Task Force, June 1999. Available at http://www.ietf.org/rfc/rfc2119.txt.

[RFC 3987]
M. Duerst, M. Suignard, "Internationalized Resource Identifiers (IRIs)", January 2005. (See *http://www.ietf.org/rfc/rfc3987.txt*.)

[XML 1.0]
Extensible Markup Language (XML) 1.0 (Third Edition), T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler, Editors. World Wide Web Consortium, 4 February 2004. This version of the XML 1.0 Recommendation is http://www.w3.org/TR/2004/REC-xml-20040204. The latest version of XML 1.0 is available at http://www.w3.org/TR/REC-xml.

[XML Namespaces]
Namespaces in XML, T. Bray, D. Hollander, and A. Layman, Editors. World Wide Web Consortium, 14 January 1999. This version of the XML Information Set Recommendation is http://www.w3.org/TR/1999/REC-xml-names-19990114. The latest version of Namespaces in XML is available at http://www.w3.org/TR/REC-xml-names.

[XML Information Set]
XML Information Set (Second Edition), J. Cowan and R. Tobin, Editors. World Wide Web Consortium, 4 February 2004. This version of the XML Information Set Recommendation is http://www.w3.org/TR/2004/REC-xml-infoset-20040204/. The latest version of XML Information Set is available at http://www.w3.org/TR/xml-infoset.

[XML Schema Structures]
XML Schema Part 1: Structures Second Edition, H. Thompson, D. Beech, M. Maloney, and N. Mendelsohn, Editors. World Wide Web Consortium, 28 October 2004. This version of the XML Schema Part 1 Recommendation is http://www.w3.org/TR/2004/REC-xmlschema-1-20041028. The latest version of XML Schema Part 1 is available at http://www.w3.org/TR/xmlschema-1.

[XML Schema Datatypes]
XML Schema Part 2: Datatypes Second Edition, P. Byron and A. Malhotra, Editors. World Wide Web Consortium, 28 October 2004. This version of the XML Schema Part 2 Recommendation is http://www.w3.org/TR/2004/REC-xmlschema-2-20041028. The latest version of XML Schema Part 2 is available at http://www.w3.org/TR/xmlschema-2.

[SOAP 1.2 Part 1: Messaging Framework]
SOAP Version 1.2 Part 1: Messaging Framework, M. Gudgin, M. Hadley, N. Mendelsohn, J-J. Moreau, H. Frystyk Nielsen, Editors. World Wide Web Consortium, 24 June 2003. This version of the "SOAP Version 1.2 Part 1: Messaging Framework" Recommendation is http://www.w3.org/TR/2003/REC-soap12-part1-20030624/. The latest version of "SOAP Version 1.2 Part 1: Messaging Framework" is available at http://www.w3.org/TR/soap12-part1/.

[WSDL 1.1]
E. Christensen, et al, *Web Services Description Language (WSDL) 1.1*, March 2001.

[WS-Security]
OASIS, *Web Services Security: SOAP Message Security*, March 2004.

A. Acknowledgements (Non-Normative)

This document is the work of the W3C Web Service Addressing Working Group.

Members of the Working Group are (at the time of writing, and by alphabetical order): Abbie Barbir (Nortel Networks), Andreas Bjärlestam (ERICSSON), Dave Chappell (Sonic Software), Eran Chinthaka (WSO2), Francisco Curbera (IBM Corporation), Glen Daniels (Sonic Software), Vikas Deolaliker (Sonoa

Systems, Inc.), Paul Downey (BT), Jacques Durand (Fujitsu Limited), Robert Freund (Hitachi, Ltd.), Marc Goodner (Microsoft Corporation), Arun Gupta (Sun Microsystems, Inc.), Hugo Haas (W3C/ERCIM), Marc Hadley (Sun Microsystems, Inc.), David Hull (TIBCO Software, Inc.), Yin-Leng Husband (HP), David Illsley (IBM Corporation), Anish Karmarkar (Oracle Corporation), Paul Knight (Nortel Networks), Philippe Le Hégaret (W3C/MIT), Amelia Lewis (TIBCO Software, Inc.), Bozhong Lin (IONA Technologies, Inc.), Mark Little (JBoss Inc.), Jonathan Marsh (Microsoft Corporation), Jeff Mischkinsky (Oracle Corporation), Nilo Mitra (ERICSSON), Eisaku Nishiyama (Hitachi, Ltd.), Ales Novy (Systinet Inc.), David Orchard (BEA Systems, Inc.), Gilbert Pilz (BEA Systems, Inc.), Tony Rogers (Computer Associates), Tom Rutt (Fujitsu Limited), Davanum Srinivas (WSO2), Jiri Tejkl (Systinet Inc.), Mike Vernal (Microsoft Corporation), Steve Vinoski (IONA Technologies, Inc.), Katy Warr (IBM Corporation), Pete Wenzel (Sun Microsystems, Inc.), Steve Winkler (SAP AG), Ümit Yalçinalp (SAP AG), Prasad Yendluri (webMethods, Inc.).

Previous members of the Working Group were: Lisa Bahler (SAIC - Telcordia Technologies), Rebecca Bergersen (IONA Technologies, Inc.), Ugo Corda (Sun Microsystems, Inc.), Michael Eder (Nokia), Yaron Goland (BEA Systems, Inc.), Marc Goodner (SAP AG), Martin Gudgin (Microsoft Corporation), Mark Nottingham (BEA Systems, Inc.), Mark Peel (Novell, Inc.), Harris Reynolds (webMethods, Inc.), Rich Salz (IBM Corporation), Davanum Srinivas (Computer Associates), Greg Truty (IBM Corporation).

The people who have contributed to discussions on public-ws-addressing@w3.org are also gratefully acknowledged.

B. Compatibility of [action] with previous versions of WS-Addressing (Non-Normative)

This section describes strategies for choosing [action] values consistent between this specification and the WS-Addressing Member Submission published 10 August 2004 (hereafter called "2004-08"). The wsa200408 namespace prefix below refers to the "http://schemas.xmlsoap.org/ws/2004/08/addressing" namespace defined in the 2004-08 version.

The WS-Addressing 1.0 [action] property, which identifies the semantics implied by a message, is semantically equivalent to the [action] message information header defined in the 2004-08 version. Authors are therefore advised to use the same value for 1.0 [action] and 2004-08 [action].

However, when describing services in WSDL, the namespace of the Action attribute used to associate values with WSDL operations differs in the two versions (wsaw:Action versus wsa200408:Action), and the default action pattern in WS-Addressing 1.0 differs in two respects from that in the 2004-08 version: the [delimiter] can be either "/" or ":" in 1.0 while in 2004-08 it is always "/", and the default action pattern for faults is closer to that of other messages instead of a constant URI.

If a default action pattern is desired, this specification recommends the 1.0 default action pattern. The 200408 [action] can be made consistent with the 1.0 default by:

1. specifying wsa200408:Action explicitly when the targetNamespace is a URN, and

2. specifying wsa200408:Action explicitly when the message is a fault.

If the targetNamespace is a URN, it is not advisable to use the 2004-08 default action pattern, as it leads to malformed IRIs. If the targetNamespace is not a URN, and the 2004-08 default action pattern is in use, the 1.0 [action] value can be made consistent by:

1. specifying wsaw:Action explicitly when the message is a fault.

C. Change Log (Non-Normative)

C.1 Changes Since Third Working Draft

Date	Editor	Description
2005-11-22 @ 21:29	mhadley	Added resolution to issue 63, new subsections describing impacts of extension elements on WSDL 2.0 component model
2005-11-07 @ 07:08	mhadley	Added resolution to issue 65, [action] defaults to same as SOAPAction in absence of wsaw:Action
2005-11-07 @ 06:44	mhadley	Updated resolution to issues 56, 57
2005-10-31 @ 20:35	mhadley	Updated UsingAddressing section to move some dense text into a simpler tabular form
2005-10-31 @ 20:12	mhadley	Added resolution to issues 56 and 57, added new top level section that describes how MAP values are derived from WSDL for [destination], [action] and [reference properties]
2005-10-24 @ 01:50	trogers	Added appendix on action compatibility with 200408 version (resolving i64)
2005-10-17 @ 18:44	mhadley	Added namesapce change policy
2005-10-11 @ 03:16	trogers	Incorporated the resolution of i61.
2005-10-10 @ 20:20	mhadley	Fixed type in example fault action URI. Added clarification that WSDL 1.1 material is included for backwards compatibility only
2005-09-15 @ 19:16	mhadley	Added resolution to issue 62 - changed Fault: to [delimiter]Fault[delimiter] in default action for WSDL 1.1 faults
2005-09-15 @ 19:09	mhadley	Added resolution to issue 20 - noted that inclusion of InterfaceName or @EndpointName in an EPR makes the EPR specific to the identified interface or endpoint respectively

2005-09-15 @ 18:47	mhadley	Added resolution to issue 17 - noted that action fulfils WSDL best practice for unique message signatures
2005-05-25 @ 21:40	mhadley	Added new section in changelog to account for previous draft publication
2005-05-18 @ 19:42	mhadley	Added lc53 resolution - expanded MAP to message addressing property and fixed editorial glitch
2005-05-18 @ 19:22	mhadley	Added lc47 resolution - fixed URL in WSDL 2.0 biblio entry
2005-04-22 @ 22:37	mhadley	Added issue 21 resolution

C.2 Changes Since Second Working Draft

Date	Editor	Description
2005-03-21 @ 23:15	mgudgin	Moved sentence on WSDL 2.0/WSDL 1.1 from Section 1.2 to Section 1
2005-03-10 @ 03:40	mhadley	Incorporated additional editorial fixes from J. Marsh.
2005-03-10 @ 02:06	mhadley	Incorporated editorial fixes from J. Marsh.
2005-03-02 @ 21:22	mhadley	Fixed some problems with use of wsdli:wsdlLocation.
2005-03-01 @ 13:33	mhadley	Changed MUST to SHOULD in section 2.2 wrt matching port name
2005-02-28 @ 22:08	mhadley	Added resolution to issues 24 and 26
2005-02-27 @ 19:42	mhadley	Changed URI to IRI where appropriate.
2005-02-23 @ 16:11	mhadley	Incorporated resolution to issue 17b
2005-02-15 @ 23:19	mhadley	Added resolution to issue 45

C.3 Changes Since First Working Draft

Date	Editor	Description
2005-02-01 @ 19:49	mhadley	Removed several occurances of the word ’identify’ when used with endpoint references. Replaced with ’reference’ or ’address’ as appropriate.
2005-01-25 @ 22:23	mhadley	Added descriptive text for wsa:Action attribute. Fixed references to WSDL 1.1 to be more explicit version-wise.
2005-01-24 @ 10:12	mgudgin	Incorporated resolution of i034 and i035; default action URI for WSDL 2.0 and default action URI for faults. All edits in section 3
2005-01-18 @ 04:01	mgudgin	Modified text in Section 2 WRT closing issue i020
2004-12-16 @ 18:20	mhadley	Added resolution to issue 19 - WSDL version neutrality
2004-12-16 @ 16:50	mhadley	Added issue 33 resolution
2004-12-14 @ 20:10	mhadley	Switched back to edcopy formatting
2004-12-14 @ 20:02	mhadley	Enhanced auto-changelog generation to allow specification of data ranges for logs. Split change log to show changes between early draft and first working draft and changes since first working draft.
2004-12-14 @ 18:13	mhadley	Added resolutions for issues 12 (EPR lifecycle), 37 (relationship from QName to URI) and 39 (spec name versioning)

C.4 Changes Since Submission

Date	Editor	Description
2004-12-04 @ 02:04	mgudgin	Added text to section on WSDL MEPs per resolution of Issue i003
2004-11-23 @ 21:38	mhadley	Updated titles of examples. Fixed table formatting and references. Replaced uuid URIs with http URIs in examples. Added document status.
2004-11-11 @ 18:31	mgudgin	Added some TBD sections
2004-11-07 @ 02:03	mhadley	Second more detailed run through to separate core, SOAP and WSDL document contents. Removed dependency on WS-Policy. Removed references to WS-Trust and WS-SecurityPolicy
2004-11-02 @ 21:45	mhadley	Replaced hardcoded change log with one generated dynamically from CVS
2004-10-28 @ 18:09	mhadley	Fixed typo in abstract
2004-10-28 @ 17:05	mhadley	Initial cut of separating specification into core, soap and wsdl