Web Services Addressing 1.0 - Metadata

W3C Working Draft 27 June 2007

This version:
http://www.w3.org/TR/2007/WD-ws-addr-metadata-20070627
Latest version:
http://www.w3.org/TR/ws-addr-metadata
Previous version:
http://www.w3.org/TR/2007/WD-ws-addr-metadata-20070516
Editors:
Martin Gudgin, Microsoft Corp
Marc Hadley, Sun Microsystems, Inc.
Tony Rogers, CA, Inc.
Ümit Yalçinalp, SAP AG

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

Abstract

Web Services Addressing provides transport-neutral mechanisms to address Web services and messages. Web Services Addressing 1.0 - Metadata (this document) defines how the abstract properties defined in Web Services Addressing 1.0 - Core are described using WSDL, how to include WSDL metadata in endpoint references, and how WS-Policy can be used to indicate the support of WS-Addressing by a Web service.

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 - Metadata specification for review by W3C members and other interested parties. It has been produced by the Web Services Addressing Working Group, which is part of the W3C Web Services Activity.

The Working Group plans to submit this specification for consideration as a W3C Proposed Recommendation shortly after the end of the Last Call phase. Comments on this document are invited and are to be sent to the public public-ws-addressing-comments@w3.org mailing list (public archive). Implementers are encouraged to provide feedback by **11 July 2007**.

The Working Group plans to submit this specification for consideration as a W3C Proposed Recommendation if the following exit criteria have been met:

- Two interoperable implementations of each optional and required feature of the specifications have been produced.
- The Working Group releases a test suite along with an implementation report.

The sections **2.1 Referencing WSDL Metadata from an EPR** [p.6] and **2.2 Embedding WSDL Metadata in an EPR** [p.6] in this specification, defining how to reference WSDL Metadata from an EPR and how to embed WSDL Metadata in an EPR, are considered at risk. Unless two interoperable implementations of those sections have been produced, the Working Group plans to remove the sections before moving forward.

No significant change has been made to this document since its previous version. The detailed disposition of those comments can be found in the Last Call issues list. 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.29] .

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

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.

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; 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) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

Appendices

---

1. Introduction

Web Services Addressing 1.0 - Core [*WS-Addressing Core [p.26]*] defines a set of abstract properties and an XML Infoset [*XML Information Set [p.27]*] representation of Web service endpoint references (EPRs) and to facilitate end-to-end addressing of endpoints in messages. Web Services Addressing 1.0 - Metadata (this document) defines how the abstract properties defined in Web Services Addressing 1.0 - Core are described using WSDL and how WS-Policy can be used to indicate the support of WS-Addressing by a Web service. WS-Addressing is designed to be able to work with WS-Policy 1.5 [*WS Policy 1.5 [p.27]*], WSDL 2.0 [*WSDL 2.0 [p.26]*] and also (for backwards compatibility) with WSDL 1.1 [*WSDL 1.1 [p.26]*] 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.26]*].

When describing abstract data models, this specification uses the notational convention used by the XML Infoset [*XML Information Set [p.27]*]. Specifically, abstract property names always appear in square brackets (e.g., [some property]).

When describing concrete XML schemas [*XML Schema Structures [p.27]* , *XML Schema Datatypes [p.27]*], this specification uses the notational convention of WS-Security [*WS-Security [p.28]*]. 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.27]*]).

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
wsam	http://www.w3.org/2007/05/addressing/metadata
wsoap	http://www.w3.org/ns/wsdl/soap
xs	http://www.w3.org/2001/XMLSchema
wsdl	Either http://www.w3.org/ns/wsdl or http://schemas.xmlsoap.org/wsdl/ depending on context
wsdl20	http://www.w3.org/ns/wsdl
wsdl11	http://schemas.xmlsoap.org/wsdl/
soap11	http://schemas.xmlsoap.org/wsdl/soap/
wsp	http://www.w3.org/ns/ws-policy

The working group intends to update the value of the Web Services Addressing 1.0 - Metadata 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 - Metadata 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.27]*]. WS-Addressing can be used with SOAP [*SOAP 1.2 [p.26]* , *SOAP 1.1 [p.26]*] as described in Web Services Addressing 1.0 - SOAP Binding [*WS-Addressing SOAP Binding [p.26]*]. The examples in this specification use an XML 1.0 [*XML 1.0 [p.27]*] representation but this is not a requirement.

All information items defined by this specification are identified by the XML namespace URI [*XML Namespaces [p.27]*] "http://www.w3.org/2007/05/addressing/metadata". A normative XML Schema [*XML Schema Structures [p.27]* , *XML Schema Datatypes [p.27]*] 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:

wsam:InterfaceName (0..1)

> 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.

wsam:ServiceName (0..1)

> 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.

wsam:ServiceName/@EndpointName (0..1)

> An NCName that identifies one endpoint amongst the set identified 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 "ghns:reservationInterface" at the endpoint IRI "http://greath.example.com/2004/reservation". Note the use of the WSDL[*WSDL 2.0 [p.26]*] wsdlLocation attribute.

Example 2-1. Example endpoint reference.

```
<wsa:EndpointReference
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:ghns="http://greath.example.com/2004/wsdl/resSvc">
  <wsa:Address>http://greath.example.com/2004/reservation</wsa:Address>
  <wsa:Metadata
      xmlns:wsdli="http://www.w3.org/ns/wsdl-instance"
      wsdli:wsdlLocation="http://greath.example.com/2004/wsdl/resSvc http://greath.example.com/2004/reservation.wsdl">
    <wsam:InterfaceName>ghns:reservationInterface</wsam: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 can 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://greath.example.com/2004/reservation</wsa:Address>
  <wsa:Metadata
      xmlns:wsdli="http://www.w3.org/ns/wsdl-instance"
      wsdli:wsdlLocation="http://greath.example.com/2004/wsdl/resSvc http://greath.example.com/2004/reservation.wsdl">
    <wsdl20:description
        targetNamespace="http://greath.example.com/2004/wsdl/resSvc"
        xmlns:tns="http://greath.example.com/2004/wsdl/resSvc"
        xmlns:ghns="http://greath.example.com/2004/wsdl/resSvc"
        xmlns:wsdl20="http://www.w3.org/ns/wsdl">
      <wsdl20:import namespace="http://greath.example.com/2004/wsdl/resSvc"
          location="http://greath.example.com/2004/reservation.wsdl"/>
      <wsdl20:service name="reservationService"
          interface="tns:reservationInterface">
        <wsdl20:endpoint name="reservationEndpoint"
          binding="tns:reservationSOAPBinding"
          address ="http://greath.example.com/2004/reservation"/>
      </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://greath.example.com/2004/reservation</wsa:Address>
  <wsa:Metadata>
    <wsdl11:definitions targetNamespace="http://greath.example.com/2004/wsdl/resSvc"
        xmlns:ghns="http://greath.example.com/2004/wsdl/resSvc"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns:wsdl11="http://schemas.xmlsoap.org/wsdl/">
      <wsdl11:import namespace="http://greath.example.com/2004/wsdl/resSvc"
          location="http://greath.example.com/2004/reservation.wsdl"/>
      <wsdl11:service name="reservationService">
        <wsdl11:port name="ep1" binding="ghns:reservationSOAPBinding">
          <soap:address location="http://greath.example.com/2004/reservation"/>
        </wsdl11:port>
      </wsdl11:service>
    </wsdl11:definitions>
  </wsa:Metadata>
</wsd:EndpointReference>
```

3. Indicating Use of WS-Addressing

This specification supports a mechanism for indicating, in a WSDL description, that the endpoint conforms to the WS-Addressing specification. That mechanism uses WS-Policy Framework [*WS Policy 1.5 [p.27]*].

3.1 WS-Policy Assertions

The mechanism for indicating that a binding or endpoint conforms to the WS-Addressing specification is through the use of the Web Services Policy - Framework [*WS Policy 1.5 [p.27]*] and Web Services Policy - Attachment [*WS Policy 1.5 - Attachment [p.27]*] specifications. This specification defines three policy assertions.

The `wsam:Addressing` policy assertion applies to the endpoint policy subject.

For WSDL 1.1, these assertions may be attached to `wsdl11:port` or `wsdl11:binding`. For WSDL 2.0, they may be attached to `wsdl20:endpoint` or `wsdl20:binding`. A policy expression containing the wsam:Addressing policy assertion MUST NOT be attached to a `wsdl:portType` or `wsdl20:interface`. The `wsam:Addressing` policy assertion specifies a concrete behavior whereas the `wsdl:portType` or `wsdl20:interface` is an abstract construct.

3.1.1 Addressing Assertion

The `wsam:Addressing` policy assertion is a nested policy container assertion. The meaning of this assertion, when present in a policy alternative, is that WS-Addressing is required to communicate with the subject. The `wsam:Addressing` assertion indicates that there are no restrictions on the use of WS-Addressing unless otherwise qualified by assertions in its nested policy expression. In order to indicate that the subject supports WS-Addressing but does not require its use, an additional policy alternative should be provided which does not contain this assertion; the compact authoring style for an optional policy assertion provided by WS-Policy V1.5 [*WS Policy 1.5 [p.27]*] may be used. The `wsp:Optional` attribute, as a syntactic shortcut, can be used with the `wsam:Addressing` assertion. This indicates two policy alternatives, one which contains the policy assertion, and another which does not.

The inclusion of this assertion implies support for the Web Services Addressing 1.0 - Core [*WS-Addressing Core [p.26]*] and Web Services Addressing 1.0 - SOAP Binding [*WS-Addressing SOAP Binding [p.26]*].

3.1.2 AnonymousResponses Assertion

The `wsam:AnonymousResponses` element MAY be used as a policy assertion nested within the `wsam:Addressing` assertion in accordance with the rules laid down by policy assertion nesting ([*WS Policy 1.5 [p.27]*], section 4.3.2).

The appearance of this element within the `wsam:Addressing` policy assertion indicates that the endpoint requires request messages to use response endpoint EPRs that contain the anonymous URI ("`http://www.w3.org/2005/08/addressing/anonymous`") as the value of [address]. In other words, the endpoint requires the use of anonymous responses.

The None URI ("`http://www.w3.org/2005/08/addressing/none`") may appear as the value of [address] in place of the anonymous URI; this value MUST be accepted.

3.1.3 NonAnonymousResponses Assertion

The `wsam:NonAnonymousResponses` element MAY be used as a policy assertion nested within the Addressing assertion in accordance with the rules laid down by policy assertion nesting ([*WS Policy 1.5 [p.27]*], section 4.3.2). The `wsam:NonAnonymousResponses` policy assertion MUST NOT be used in the same policy alternative as the `wsam:AnonymousResponses` policy assertion.

The appearance of this element within the `wsam:Addressing` assertion indicates that the endpoint expresses requires request messages to use response endpoint EPRs that contain something other than the anonymous URI as the value of [address]. In other words, the endpoint requires the use of non-anonymous responses. This assertion is deliberately vague; its presence indicates that some non-anonymous addresses will be accepted but doesn't constrain what such an address might look like. A receiver can still reject a request that contains an address that it doesn't understand or that requires a binding it doesn't support.

The None URI ("`http://www.w3.org/2005/08/addressing/none`") may appear as the value of [address] in place of a non-anonymous address; this value MUST be accepted.

3.1.4 Examples (Compact Form)

Example 3-1. Subject supports WS-Addressing

```
<wsp:Policy>
    <wsam:Addressing wsp:Optional="true">
        <wsp:Policy/>
    </wsam:Addressing>
</wsp:Policy>
```

Example 3-2. Subject requires WS-Addressing

```
<wsp:Policy>
    <wsam:Addressing>
        <wsp:Policy/>
    </wsam:Addressing>
</wsp:Policy>
```

Example 3-3. Subject requires WS-Addressing and requires the use of non-anonymous response EPRs

```
<wsp:Policy>
    <wsam:Addressing>
        <wsp:Policy>
            <wsam:NonAnonymousResponses/>
        </wsp:Policy>
    </wsam:Addressing>
</wsp:Policy>
```

3.1.5 Examples (Normal Form)

Example 3-4. Subject supports WS-Addressing

```
<wsp:Policy>
    <wsp:ExactlyOne>
        <wsp:All/>
        <wsp:All>
            <wsam:Addressing>
                <wsp:Policy>
                    <wsp:ExactlyOne>
                        <wsp:All/>
                    </wsp:ExactlyOne>
                </wsp:Policy>
            </wsam:Addressing>
        </wsp:All>
    </wsp:ExactlyOne>
</wsp:Policy>
```

Example 3-5. Subject requires WS-Addressing

```
<wsp:Policy>
    <wsp:ExactlyOne>
        <wsp:All>
            <wsam:Addressing>
                <wsp:Policy>
                    <wsp:ExactlyOne>
                        <wsp:All/>
                    </wsp:ExactlyOne>
                </wsp:Policy>
            </wsam:Addressing>
        </wsp:All>
    </wsp:ExactlyOne>
</wsp:Policy>
```

Example 3-6. Subject requires WS-Addressing and requires the use of non-anonymous response EPRs

```
<wsp:Policy>
    <wsp:ExactlyOne>
        <wsp:All>
            <wsam:Addressing>
                <wsp:Policy>
                    <wsp:ExactlyOne>
                        <wsp:All>
                            <wsam:NonAnonymousResponses/>
                        </wsp:All>
                    </wsp:ExactlyOne>
                </wsp:Policy>
            </wsam:Addressing>
        </wsp:All>
    </wsp:ExactlyOne>
</wsp:Policy>
```

3.1.6 Finding Compatible Policies

When a client is looking for an endpoint with compatible policy, one common method used is to take the policy intersection between the policy which the client is looking for, and the policy asserted in the WSDL document; a non-empty intersection is sought. The policy used by the client must be written carefully to avoid unexpected results. This is most obvious when the client is not looking for explicit support of a particular kind of response; failing to take care could mean missing a compatible policy.

Example 3-7. Client looking for an endpoint which supports Addressing, and which supports anonymous responses

```
<wsp:Policy>
    <wsp:ExactlyOne>
        <wsp:All>
            <wsam:Addressing>
                <wsp:Policy>
                    <wsp:ExactlyOne>
                        <wsp:All>
                            <AnonymousResponses Optional=”true”/>
                        </wsp:All>
                    </wsp:ExactlyOne>
                </wsp:Policy>
            </wsam:Addressing>
        </wsp:All>
    </wsp:ExactlyOne>
</wsp:Policy>
```

Example 3-8. Client looking for an endpoint which supports Addressing, and does not require support for responses (will intersect with anything)

```
<wsp:Policy>
    <wsp:ExactlyOne>
        <wsp:All>
            <wsam:Addressing> <-- supports all response types -->
                <wsp:Policy>
                </wsp:Policy>
            </wsam:Addressing>
        </wsp:All>
        <wsp:All>
            <wsam:Addressing> <-- requires Anonymous responses -->
                <wsp:Policy>
                    <wsp:ExactlyOne>
                        <wsp:All>
                            <AnonymousResponses />
                        </wsp:All>
                    </wsp:ExactlyOne>
                </wsp:Policy>
            </wsam:Addressing>
        </wsp:All>
        <wsp:All>
            <wsam:Addressing> <-  requires nonAnonymous responses -->
                <wsp:Policy>
                    <wsp:ExactlyOne>
                        <wsp:All>
```

```
                    <NonAnonymousResponses />
                </wsp:All>
            </wsp:ExactlyOne>
        </wsp:Policy>
    </wsam:Addressing>
  </wsp:All>
 </wsp:ExactlyOne>
</wsp:Policy>
```

For more detailed descriptions of the use of wsp:Optional, wsp:Ignorable, and strict and lax intersection, please refer to the WS-Policy Primer [*WS Policy 1.5 - Primer [p.27]*].

4. Specifying Message Addressing Properties in WSDL

This section describes how the values of certain message addressing properties can 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 Extending WSDL Endpoints with an EPR

A wsdl20:endpoint or wsdl11:port element MAY be extended using a child wsa:EndpointReference element. When extended this way, the [address] property of the child EPR MUST match the {address} property of the endpoint component (WSDL 2.0) or the address value provided by the relevant port extension (WSDL 1.1). For example, in a SOAP 1.1 port described using WSDL 1.1, the location attribute of a soap11:address element (if present) would have the same value as the wsa:Address child element of the wsa:EndpointReference element.

4.1.1 WSDL 2.0 Component Model Changes

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

- A property of the Endpoint component, named {endpoint reference}. This property is of type wsa:EndpointReference, with a cardinality of 1. The property has the value of the wsa:EndpointReference element used as a child of wsdl20:endpoint, if any. If no such extension exists, this property is absent.

4.2 Destination

The value of the [destination] message addressing property for a message sent to an endpoint typically matches the value of the {address} property of the endpoint component (WSDL 2.0) or the address value (if any) 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. For an endpoint or port extended with an EPR (see **4.1 Extending WSDL Endpoints with an EPR** [p.12]), the value is provided by the [address] property of the EPR.

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.

4.3 Reference Parameters

When a wsa:EndpointReference element is present in a wsdl20:endpoint or a wsdl11:port element (see **4.1 Extending WSDL Endpoints with an EPR** [p.12]), the value of the [reference parameters] message addressing property for a message sent to an endpoint MUST include the contents of the wsa:ReferenceParameters element, if one exists within that EPR.

4.4 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 **4.4.1 Explicit Association** [p.13] ; action defaulting (where a unique value for the [action] property is automatically generated) is described in section **4.4.4 Default Action Pattern for WSDL 1.1** [p.17] for WSDL 1.1 and section **4.4.2 Default Action Pattern for WSDL 2.0** [p.14] 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.26]* . The [action] property provides a mechanism to fulfill that best practice.

4.4.1 Explicit Association

WS-Addressing defines a global attribute, wsam:Action, that can 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 wsam:Action attribute on a WSDL input element where a non-empty SOAPAction value is specified, the value of the [action] property for the input message is the value of the SOAPAction specified. If the wsam:Action attribute is absent, and SOAPAction is not specified, or is empty, then the default pattern is used. Note that the SOAPAction value is not required to be an absolute IRI, but the [action] property is required to be an absolute IRI; if WS-Addressing is required (the wsam:Addressing assertion is present), wsam:Action is not specified, and the SOAPAction value is not empty or an absolute IRI, then the document MUST be considered invalid. Web Services Addressing 1.0 - SOAP Binding[*WS-Addressing SOAP Binding [p.26]*] specifies restrictions on the relationship between the values of [action] and SOAPAction for SOAP 1.1 and SOAP 1.2.

The inclusion of wsam:Action without the inclusion of the wsam:Addressing assertion has no normative intent and is only informational. In other words, the inclusion of wsam: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 the wsam:Addressing assertion. Other specifications defining the value of [action] are under no constraint to be consistent with wsam:Action.

For example consider the following WSDL excerpt:

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

```
<description targetNamespace="http://greath.example.com/2004/schemas/resSvc" ...>
  ...
  <interface name="reservationInterface">
    <operation name="opCheckAvailability" pattern="http://www.w3.org/ns/wsdl/in-out">
      <input element="tns:checkAvailability" messageLabel="In"
            wsam:Action="http://greath.example.com/2004/wsdl/resSvc/opCheckAvailability"/>
      <output element="tns:checkAvailabilityResponse" messageLabel="Out"
            wsam:Action="http://greath.example.com/2004/wsdl/resSvc/opCheckAvailabilityResponse"/>
    </operation>
  </interface>
  ...
</description>
```

The action for the input of the opCheckAvailability operation within the reservationInterface is explicitly defined to be http://greath.example.com/2004/wsdl/resSvc/opCheckAvailability. The action for the output of this same operation is http://greath.example.com/2004/wsdl/resSvc/opCheckAvailabilityResponse.

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

```
<definitions targetNamespace="http://greath.example.com/2004/schemas/resSvc" ...>
  ...
  <portType name="reservationInterface">
    <operation name="opCheckAvailability">
      <input message="tns:checkAvailability"
            wsam:Action="http://greath.example.com/2004/wsdl/resSvc/opCheckAvailability"/>
      <output message="tns:checkAvailabilityResponse"
            wsam:Action="http://greath.example.com/2004/wsdl/resSvc/opCheckAvailabilityResponse"/>
    </operation>
  </portType>
  ...
</definitions>
```

The action for the input of the opCheckAvailability operation within the reservationInterface port type is explicitly defined to be http://greath.example.com/2004/wsdl/resSvc/opCheckAvailability. The action for the output of this same operation is http://greath.example.com/2004/wsdl/resSvc/opCheckAvailabilityResponse.

4.4.2 Default Action Pattern for WSDL 2.0

In the absence of an explicitly specified value for the [action] property (see section **4.4.1 Explicit Association** [p.13]), 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][operation name][direction token][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 might 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/ns/wsdl/in-only", "http://www.w3.org/ns/wsdl/robust-in-only", "http://www.w3.org/ns/wsdl/out-only", or "http://www.w3.org/ns/wsdl/robust-out-only".
- "Request" where the operation's {message exchange pattern} is "http://www.w3.org/ns/wsdl/in-out" or "http://www.w3.org/ns/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/ns/wsdl/out-in" or "http://www.w3.org/ns/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/ns/wsdl/in-out" or "http://www.w3.org/ns/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/ns/wsdl/out-in", or "http://www.w3.org/ns/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.

```
<description targetNamespace="http://greath.example.com/2004/wsdl/resSvc" ...>
  ...
  <interface name="reservationInterface">
    <operation name="opCheckAvailability" pattern="http://www.w3.org/ns/wsdl/in-out">
      <input element="tns:checkAvailability" messageLabel="In"/>
      <output element="tns:checkAvailabilityResponse" messageLabel="Out"/>
    </operation>
  </interface>
  ...
</definitions>
```

[targetNamespace] = http://greath.example.com/2004/wsdl/resSvc

[interface name] = reservationInterface

[operation name] = opCheckAvailability

[direction token] for input is Request

[direction token] for output is Response

Applying the patterns above with these values we have:

input action =
http://greath.example.com/2004/wsdl/resSvc/reservationInterface/opCheckAvailabilityRequest

output action =
http://greath.example.com/2004/wsdl/resSvc/reservationInterface/opCheckAvailabilityResponse

fault action for a fault named AvailabilityNotAvailableFault =
http://greath.example.com/2004/wsdl/resSvc/reservationInterface/opCheckAvailabilityResponse/AvailabilityNotAvailableFault

4.4.3 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 Operation InFault/OutFault components named {action}. The property is of type xs:anyURI. The property value is either explicitly specified, as described in section **4.4.1 Explicit Association** [p.13] , or the default value computed following the rules from section **4.4.2 Default Action Pattern for WSDL 2.0** [p.14] .

4.4.4 Default Action Pattern for WSDL 1.1

A default pattern is also defined for backwards compatibility with WSDL 1.1. In the absence of an explicitly specified value for the [action] property (see section **4.4.1 Explicit Association** [p.13]), 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 might 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).

[operation name]

is the {name} of the operation.

[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://greath.example.com/2004/wsdl/resSvc" ...>
  ...
  <portType name="reservationInterface">
    <operation name="opCheckAvailability">
      <input message="tns:checkAvailability" name="CheckAvailability"/>
      <output message="tns:checkAvailabilityResponse" name="Availability"/>
          <fault message="tns:InvalidDate" name="InvalidDate"/>
    </operation>
  </portType>
  ...
</definitions>
```

[targetNamespace] = http://greath.example.com/2004/wsdl/resSvc

[port type name] = reservationInterface

[input name] = CheckAvailability

[output name] = CheckAvailabilityResponse

[fault name] = InvalidDate

Applying the pattern above with these values we have:

input action = http://greath.example.com/2004/wsdl/resSvc/reservationInterface/CheckAvailability

output action = http://greath.example.com/2004/wsdl/resSvc/reservationInterface/Availability

fault action =
http://greath.example.com/2004/wsdl/resSvc/reservationInterface/opCheckAvailability/Fault/InvalidDate

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://greath.example.com/2004/wsdl/resSvc" ...>
  ...
  <portType name="reservationInterface">
    <operation name="opCheckAvailability">
      <input message="tns:checkAvailability"/>
      <output message="tns:checkAvailabilityResponse"/>
    </operation>
  </portType>
  ...
</definitions>
```

[targetNamespace] = http://greath.example.com/2004/wsdl/resSvc

[port type name] = reservationInterface

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] = opCheckAvailabilityRequest

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

[output name] = opCheckAvailabilityResponse

Applying the pattern above with these values we have:

input action =
http://greath.example.com/2004/wsdl/resSvc/reservationInterface/opCheckAvailabilityRequest

output action =
http://greath.example.com/2004/wsdl/resSvc/reservationInterface/opCheckAvailabilityResponse

5. WS-Addressing and WSDL Message Exchange Patterns

This section describes which of the core message properties are mandatory 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 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 could be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but could be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but could 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 could 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 could 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 could 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 could be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but could be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but could 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.19] 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.20] apply to this MEP also.

5.2 WSDL 2.0 Message Exchange Patterns

This section describes which of the core message properties are mandatory for messages in the various MEPs defined by WSDL 2.0 [*WSDL 2.0 Adjuncts [p.26]*].

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 could be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but could be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but could 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 could 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 could be included to facilitate longer running message exchanges.
[reply endpoint]	N*	Intended receiver for replies to this message. Unused in this MEP, but could 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 could be included to facilitate longer running message exchanges.

* Note that at least one of [fault endpoint] or [reply endpoint] is required for this MEP, so that a fault can be sent if necessary.

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 could be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but could be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but could 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 could 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 could 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 could be included to facilitate longer running message exchanges.
[reply endpoint]	N	Intended receiver for replies to this message. Unused in this MEP, but could be included to facilitate longer running message exchanges.
[fault endpoint]	N	Intended receiver for faults related to this message. Unused in this MEP, but could 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.23] 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.21] 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.22] 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.23] 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.23] apply to this MEP also

6. Conformance

An endpoint reference whose wsa:Metadata element has among its children the elements defined in **2.1 Referencing WSDL Metadata from an EPR** [p.6] conforms to this specification if it obeys the structural constraints defined in that section.

A WSDL description conforms to this specification when it incorporates directly or indirectly the **3.1 WS-Policy Assertions** [p.8] marker, and obeys the structural constraints defined in section **3. Indicating Use of WS-Addressing** [p.8] appropriate to that marker, and those defined in section **4.4 Action** [p.13] .

An endpoint conforms to this specification if it has a conformant WSDL description associated with it, and receives and emits messages in accordance with the constraints defined in sections **4. Specifying Message Addressing Properties in WSDL** [p.12] and **5. WS-Addressing and WSDL Message Exchange Patterns** [p.19] .

7. References

7.1 Normative

[IETF RFC 2119]
Key words for use in RFCs to Indicate Requirement Levels, S. Bradner, Author. Internet Engineering Task Force, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt

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

[SOAP 1.1]
Simple Object Access Protocol (SOAP) 1.1, D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H Frystyk Nielsen, S. Thatte, D. Winer, Editors. W3C Member Submission, 8 May 2000.

[SOAP 1.2]
SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), M. Gudgin, et al., Editors. World Wide Web Consortium, 24 June 2003, revised 27 April 2007. This version of the "SOAP Version 1.2 Part 1: Messaging Framework" Recommendation is http://www.w3.org/TR/2007/REC-soap12-part1-20070427/. The latest version of "SOAP Version 1.2 Part 1: Messaging Framework" is available at http://www.w3.org/TR/soap12-part1/.

[WS-Addressing Core]
Web Services Addressing 1.0 - Core, M. Gudgin, M. Hadley, and T. Rogers, Editors. World Wide Web Consortium, 9 May 2006. This version of the WS-Addressing Core Recommendation is http://www.w3.org/TR/2006/REC-ws-addr-core-20060509. The latest version of WS-Addressing Core is available at http://www.w3.org/TR/ws-addr-core.

[WS-Addressing SOAP Binding]
Web Services Addressing 1.0 - SOAP Binding, M. Gudgin, M. Hadley, and T. Rogers, Editors. World Wide Web Consortium, 9 May 2006. This version of the WS-Addressing Core Recommendation is http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509. The latest version of WS-Addressing SOAP Binding is available at http://www.w3.org/TR/ws-addr-soap.

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

[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, 26 June 2007. This version of the "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language" Recommendation is available is available at http://www.w3.org/TR/2007/REC-wsdl20-20070626. The latest version of "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language" 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, 26 June 2007. This version of the "Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts" Recommendation is available at http://www.w3.org/TR/2007/REC-wsdl20-adjuncts-20070626. The latest version of "Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts" is available at http://www.w3.org/TR/wsdl20-adjuncts.

[WS Policy 1.5]
Web Services Policy 1.5 - Framework, Asir S Vedamuthu, et al., Editors. World Wide Web Consortium, 5 June 2007. This version of the WS-Policy Framework is http://www.w3.org/TR/2007/CR-ws-policy-20070605. The latest version of WS Policy Framework is available at http://www.w3.org/TR/ws-policy

[WS Policy 1.5 - Attachment]
Web Services Policy 1.5 - Attachment, Asir S Vedamuthu, et al., Editors. World Wide Web Consortium, 5 June 2007. This version of the WS-Policy Attachment is http://www.w3.org/TR/2007/CR-ws-policy-attach-20070605. The latest version of WS Policy Attachment is available at http://www.w3.org/TR/ws-policy-attach

[XML 1.0]
Extensible Markup Language (XML) 1.0 (Fourth Edition), T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler, Editors. World Wide Web Consortium, 10 February 1998, revised 16 August 2006. This version of the XML 1.0 Recommendation is http://www.w3.org/TR/2006/REC-xml-20060816. The latest version of XML 1.0 is available at http://www.w3.org/TR/xml.

[XML Namespaces]
Namespaces in XML 1.0 (Second Edition), T. Bray, D. Hollander, A. Layman, and R. Tobin, Editors. World Wide Web Consortium, 14 January 1999, revised 16 August 2006. This version of Namespaces in XML 1.0 Recommendation is http://www.w3.org/TR/2006/REC-xml-names-20060816/. The latest version of Namespaces in XML is available at http://www.w3.org/TR/xml-names.

[XML Information Set]
XML Information Set (Second Edition), J. Cowan and R. Tobin, Editors. World Wide Web Consortium, 24 October 2001, revised 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, 2 May 2001, revised 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, 2 May 2001, revised 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.

7.2 Informative

[WS Policy 1.5 - Primer]
Web Services Policy 1.5 - Primer, Asir S Vedamuthu, et al., Editors. World Wide Web Consortium, 5 June 2007. This version of the WS-Policy Primer is http://www.w3.org/TR/2007/WD-ws-policy-primer-20070605. The latest version of WS Policy Primer is available at http://www.w3.org/TR/ws-policy-primer

[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), 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), David Hull (TIBCO Software, Inc.), Yin-Leng Husband (HP), David Illsley (IBM Corporation), Ram Jeyaraman (Microsoft 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.), Jeganathan Markandu (Nortel Networks), 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.), Rama Pulavarthi (Sun Microsystems, Inc.), Alain Regnier (Ricoh Company, Ltd.), Tony Rogers (CA), Tom Rutt (Fujitsu Limited), Davanum Srinivas (WSO2), Jiri Tejkl (Systinet Inc.), Katy Warr (IBM Corporation), 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.), Vladislav Bezrukov (SAP AG), Dave Chappell (Sonic Software), Ugo Corda (Sun Microsystems, Inc.), Michael Eder (Nokia), Yaron Goland (BEA Systems, Inc.), Martin Gudgin (Microsoft Corporation), Arun Gupta (Sun Microsystems, Inc.), Hugo Haas (W3C), Marc Hadley (Sun Microsystems, Inc), Jonathan Marsh (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), Mike Vernal (Microsoft Corporation), Steve Vinoski (IONA Technologies, Inc.), Pete Wenzel (Sun Microsystems, Inc.).

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 (wsam: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 wsam:Action explicitly when the message is a fault.

C. Change Log (Non-Normative)

C.1 Changes Since Candidate Recommendation Draft

Date	Editor	Description
2007-06-19 @ 14:18	plehegar	Fixed [operation name] in default action pattern for WSDL 1.1
2007-06-19 @ 14:13	plehegar	Updatd references
2007-06-19 @ 04:22	trogers	LC141: change default action pattern for faults in WSDL 2.0
2007-06-19 @ 00:03	trogers	LC143: change statement of how Policy’s optional attribute works to a reference to WS-Policy
2007-06-18 @ 23:40	trogers	Included [operation name] in the definitions in default action pattern for WSDL 1.1
2007-05-17 @ 21:08	plehegar	Updated prevloc following the publication
2007-05-16 @ 19:29	plehegar	Updated latest versions links for XML and Namespaces
2007-05-15 @ 15:13	plehegar	Updated Tony’s affiliation and references

2007-05-14 @ 20:52	plehegar	New namespace for the specification, policy, and WSDL
2007-04-26 @ 12:34	plehegar	lc134 with "Add to Section 3 the language " This assertion implies support for ws-addr core and soap binding."
2007-01-31 @ 15:23	plehegar	Minor tweaks in the abstract
2007-01-31 @ 13:18	plehegar	Removed extra whitespaces, added missing whitespaces, fixed introduction to include mention of WS-Policy, changed prefix from "ws-addr-wsdl" to "ws-addr-metadata", latest version is CR of May 29, changed status back to WD, updated references (new editions of XML 1.0 and XML Namespaces, new versions of WSDL 2.0, new policy primer)
2007-01-31 @ 10:57	trogers	Removed the last traces of UsingAddressing
2007-01-30 @ 10:44	trogers	Adjusted definitions of response assertions, Replaced section 3.1.4 (moved to 3.1.6 and changed content)
2007-01-29 @ 09:37	trogers	Corrected the captions on examples 3-5 and 3-10
2007-01-22 @ 22:58	trogers	Corrected message label case in example 4-5 - editorial
2007-01-22 @ 22:45	trogers	Implemented editorial changes resulting from 23Jan telecon. Implemented CR40 and CR42.
2007-01-14 @ 20:31	trogers	removed UsingAddressing and SOAP module, updated conformance to suit, added None URI as acceptable
2007-01-10 @ 11:26	trogers	Changed the namespace and namespace prefix for this document Corrected introduction and conformance section
2007-01-05 @ 14:20	trogers	Implemented the resolutions of CR33 and CR44: policy assertions for using addressing and anon/non-anon responses
2007-01-03 @ 12:28	trogers	Implemented the resolution of CR30 - SOAPAction not empty or absolute IRI makes the document invalid
2007-01-03 @ 11:38	trogers	Implemented CR26 and CR28 about the rules for populating the [action] property.

C.2 Changes Since Last Call Working Draft

Date	Editor	Description

2006-05-04 @ 12:33	mhadley	Split the references into normative and informative, fixed a few editorial glitches
2006-04-28 @ 15:09	mhadley	Added new change log section for LC issues
2006-04-28 @ 15:04	mhadley	Incorporated resolution to issue lc132 - reworked section 4 to allow use of EPRs as WSDL endpoint/port extensions
2006-04-28 @ 13:40	trogers	Implemented the resolution of LC131, simplifying table 3.1 to remove discussion of UsingAddressing not present.
2006-04-28 @ 13:25	trogers	Implemented the resolution of LC129, removing the default for wsaw:Anonymous
2006-04-28 @ 13:09	trogers	Implemented LC124, adding Conformance section.
2006-04-26 @ 15:34	mhadley	Added resolution of issue lc122 - added (n..m) notation to wsaw:InterfaceName, wsaw:ServiceName and wsaw:ServiceName/@EndpointName descriptions
2006-04-26 @ 15:28	mhadley	Added resolution of issue lc123 - changed all the examples to be based on the one used in the WSDL 2.0 primer
2006-04-17 @ 10:27	trogers	Removed MUST from section 4.1 concerning the value of [destination] (LC130)
2006-04-17 @ 10:14	trogers	Marking UsingAddressing using <el> tag to show that it is not a typo in heading 3.1 (LC126)
2006-04-17 @ 10:05	trogers	Added the class of product specification to the Abstract (LC125)
2006-04-17 @ 09:46	trogers	Applied the changes required for LC120 - typo in intro and correcting wsa:Action/wsaw:Action.
2006-04-17 @ 09:34	trogers	Applied the changes required for LC119.
2006-04-17 @ 08:42	trogers	Changed the {reference parameters} property from REQUIRED to OPTIONAL in the component model. This completes LC116.
2006-03-27 @ 19:48	mhadley	Used alternate words instead of lowercase RFC2119 terms
2006-03-20 @ 15:05	mhadley	Fixed a typo in example generated fault action
2006-03-15 @ 22:56	trogers	Implemented the resolution of LC116: added section describing the {reference parameters} property.

2006-03-13 @ 13:30	trogers	Added the resolution of LC113: clarifying section 3.3 WSDL SOAP module.
2006-03-13 @ 13:19	trogers	Added the resolution of LC111: clarifying the {addressing required} property.
2006-03-13 @ 13:03	trogers	Altered changelog limit from start of 2006 to end of 2006.
2006-03-13 @ 12:59	trogers	Added resolution of LC109: specify that at least one of reply or fault endpoint is required on Robust In-Only
2006-03-03 @ 14:10	mhadley	Fixed editor list in references
2006-03-03 @ 13:48	mhadley	Added resolution to LC115 - definition to description for WSDL 2.0
2006-03-03 @ 13:45	mhadley	Added resolution to LC114 - typos
2006-02-22 @ 14:22	mhadley	Fixed a typo: ’by by’ to ’by’

C.3 Changes Since Third Working Draft

Date	Editor	Description
2006-02-13 @ 20:15	mhadley	Removed ed notes
2006-02-13 @ 16:56	mhadley	A few grammar fixes and noted that wsaw:Anonymous with a value of optional is equivalent to the default.
2006-02-13 @ 16:45	mhadley	Added resolution to issue 70, soften language on defining value of [destination] to allow runtime override.
2006-02-13 @ 15:50	mhadley	Added resolution to issue 66, explicit note that wsaw:UsingAddressing could be used outside WSDL, e.g. in a policy framework
2006-01-19 @ 20:37	mhadley	Fixed some grammar errors
2006-01-08 @ 23:14	trogers	Umit’s description of the Anonymous element added; Umit added to editor list.
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.4 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.5 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.6 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