Web Services Description Language (WSDL) Version 1.2: Bindings

W3C Working Draft 24 January 2003

This version:
http://www.w3.org/TR/2003/WD-wsdl12-bindings-20030124
Latest version:
http://www.w3.org/TR/wsdl12-bindings
Previous versions:
http://www.w3.org/TR/2002/WD-wsdl12-bindings-20020709
Editors:
Jean-Jacques Moreau, Canon
Jeffrey Schlimmer, Microsoft

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

Copyright © 2003 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, trademark, document use and software licensing rules apply.

Abstract

WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. WSDL Version 1.2 Bindings describes how to use WSDL in conjunction with SOAP 1.2 [*SOAP 1.2 Part 1: Messaging Framework [p.28]*], HTTP/1.1 GET/POST [*IETF RFC 2616 [p.28]*], and MIME [*IETF RFC 2045 [p.27]*]. This specification depends on WSDL Version 1.2 [*WSDL 1.2 [p.29]*].

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained at the W3C.

This is a W3C Working Draft of the WSDL Version 1.2 Bindings specification for review by W3C members and other interested parties.

This document has been produced as part of the W3C Web Services Activity. The authors of this document are the Web Services Description Working Group members.

This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress". This is work in progress and does not imply endorsement by, or the consensus of, either W3C or members of the Web Services Description Working Group.

For a detailed list of changes since the last publication of this document, refer to appendix **B. Part 2 Change Log** [p.30] .

Comments on this document are invited and are to be sent to public-ws-desc-comments@w3.org (public archive). It is inappropriate to send discussion emails to this address. Discussion of this document takes place on the public www-ws-desc@w3.org mailing list (public archive).

Patent disclosures relevant to this specification may be found on the Working Group's patent disclosure page.

A list of current W3C Recommendations and other technical documents can be found at http://www.w3.org/TR.

Short Table of Contents

1. Introduction [p.4]
2. SOAP Binding [p.6]
3. HTTP GET and POST Binding [p.20]
4. MIME Binding [p.23]
5. References [p.27]
A. Acknowledgements [p.29] (Non-Normative)
B. Part 2 Change Log [p.30] (Non-Normative)

Table of Contents

1. Introduction [p.4]
  1.1 Notational Conventions [p.4]
2. SOAP Binding [p.6]
  2.1 Pseudo Schema (Non-Normative) [p.6]
  2.2 binding Element with binding [parent] [p.7]
    2.2.1 transportAttribute with binding [owner] [p.8]
    2.2.2 styleDefault Attribute with binding [owner] [p.8]
    2.2.3 namespaceDefault Attribute with binding [owner] [p.9]
    2.2.4 encodingStyleDefault Attribute with binding [owner] [p.9]
  2.3 module Element with binding [parent] [p.10]
    2.3.1 uri Attribute with module [owner] [p.10]
    2.3.2 required Attribute with binding [owner] [p.10]
  2.4 propertyConstraint Element with module [parent] [p.11]
    2.4.1 uri Attribute with propertyConstraint [owner] [p.11]
    2.4.2 type Attribute with propertyConstraint [owner] [p.11]

2.5 operation Element with operation [parent] [p.12]
2.5.1 style Attribute with operation [owner] [p.12]
2.5.2 soapAction Attribute with operation [owner] [p.13]
2.6 body Element with input or output [parent] [p.13]
2.6.1 namespace Attribute with body [owner] [p.14]
2.6.2 encodingStyle Attribute with body [owner] [p.14]
2.7 header Element with input or output [parent] [p.15]
2.7.1 element Attribute with header [owner] [p.16]
2.7.2 type Attribute with header [owner] [p.16]
2.7.3 localname Attribute with header [owner] [p.16]
2.7.4 namespace Attribute with header [owner] [p.17]
2.7.5 encodingStyle Attribute with header [owner] [p.17]
2.7.6 role Attribute with header [owner] [p.17]
2.8 headerfault Element with input or output [parent] [p.18]
2.9 fault Element with operation [parent] [p.19]
2.10 address Element with port [parent] [p.19]
2.10.1 location Attribute with address [owner] [p.19]
3. HTTP GET and POST Binding [p.20]
3.1 HTTP GET/POST Examples [p.20]
3.2 How the HTTP GET/POST Binding Extends WSDL [p.21]
3.3 http:address [p.22]
3.4 http:binding [p.22]
3.5 http:operation [p.22]
3.6 http:urlEncoded [p.23]
3.7 http:urlReplacement [p.23]
4. MIME Binding [p.23]
4.1 MIME Binding example [p.24]
4.2 How the MIME Binding extends WSDL [p.25]
4.3 mime:content [p.26]
4.4 mime:multipartRelated [p.27]
4.5 soap:body [p.27]
4.6 mime:mimeXml [p.27]
5. References [p.27]
5.1 Normative References [p.27]
5.2 Informative References [p.29]

Appendices

A. Acknowledgements [p.29] (Non-Normative)
B. Part 2 Change Log [p.30] (Non-Normative)
B.1 WSDL Specification Changes [p.30]
B.2 XML Schema Changes [p.31]

1. Introduction

The Web Services Description Language WSDL Version 1.2 (WSDL) [*WSDL 1.2 [p.29]*] defines an XML grammar [*XML 1.0 [p.28]*] for describing network services as collections of communication endpoints capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details involved in applications communication. WSDL 1.2 Bindings (this document) defines binding extensions for the following protocols and message formats:

- SOAP Version 1.2 [*SOAP 1.2 Part 1: Messaging Framework [p.28]*] (see **2. SOAP Binding** [p.6]).
- HTTP/1.1 GET/POST [*IETF RFC 2616 [p.28]*] (see **3. HTTP GET and POST Binding** [p.20]).
- MIME [*IETF RFC 2045 [p.27]*] (see **4. MIME Binding** [p.23]).

WSDL 1.2 Primer [*WSDL 1.2 Primer [p.29]*] is a non-normative document intended to provide an easily understandable tutorial on the features of the WSDL Version 1.2 specifications.

WSDL 1.2 [*WSDL 1.2 [p.29]*] of the WSDL specification describes the core elements of the WSDL language.

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 RFC2119 [*IETF RFC 2119 [p.28]*].

This specification uses a number of namespace prefixes throughout; they are listed in **Table 1** [p.4] . Note that the choice of any namespace prefix is arbitrary and not semantically significant (see [*XML Information Set [p.28]*]).

Table 1: Prefixes and Namespaces used in this specification

Prefix	Namespace	Notes
wsdl	"http://www.w3.org/2003/01/wsdl"	A normative XML Schema [*XML Schema Structures [p.28]*], [*XML Schema Datatypes [p.29]*] document for the "http://www.w3.org/2003/01/wsdl" namespace can be found at http://www.w3.org/2003/01/wsdl.
soap	"http://www.w3.org/2003/01/wsdl/soap12"	A normative XML Schema [*XML Schema Structures [p.28]*], [*XML Schema Datatypes [p.29]*] document for the "http://www.w3.org/2003/01/wsdl/soap12" namespace can be found at http://www.w3.org/2003/01/wsdl/soap12.
http	"http://www.w3.org/2003/01/wsdl/http"	A normative XML Schema [*XML Schema Structures [p.28]*], [*XML Schema Datatypes [p.29]*] document for the "http://www.w3.org/2003/01/wsdl/http" namespace can be found at http://www.w3.org/2003/01/wsdl/http.
mime	"http://www.w3.org/2003/01/wsdl/mime"	A normative XML Schema [*XML Schema Structures [p.28]*], [*XML Schema Datatypes [p.29]*] document for the "http://www.w3.org/2003/01/wsdl/mime" namespace can be found at http://www.w3.org/2003/01/wsdl/mime.
xsd	"http://www.w3.org/2001/XMLSchema"	Defined in the W3C XML Schema specification [*XML Schema Structures [p.28]*], [*XML Schema Datatypes [p.29]*].
xsi	"http://www.w3.org/2001/XMLSchema-instance"	Defined in the W3C XML Schema specification [*XML Schema Structures [p.28]*], [*XML Schema Datatypes [p.29]*].

Namespace names of the general form "http://example.org/..." and "http://example.com/..." represent application or context-dependent URIs [*IETF RFC 2396 [p.28]*].

This specification uses the Extended Backus-Naur Form (EBNF) as described in XML 1.0 [*XML 1.0 [p.28]*].

With the exception of examples and sections explicitly marked as "Non-Normative", all parts of this specification are normative.

2. SOAP Binding

Editorial note: JCS	20030115
The WG is actively redesigning the SOAP / HTTP binding to align with changes in SOAP 1.2.	

Editorial note: JCS	20030115
This section is not fully converted to a component model compatible with Part 1.	

WSDL includes a binding for SOAP 1.2 endpoints, which supports the specification of the following protocol specific information:

- An indication that a binding is bound to the SOAP 1.2 protocol.
- A way of specifying an address for a SOAP endpoint.
- The URI for the SOAPAction HTTP header for the HTTP binding of SOAP.

Editorial note: JCS	20030115
The SOAP 1.2 HTTP binding does not define a SOAPAction HTTP header. This is a known issue.	

- A list of definitions for Headers that are transmitted as part of the SOAP Envelope

This binding grammar is not an exhaustive specification since the set of SOAP bindings is evolving. Nothing precludes additional SOAP bindings to be derived from portions of this grammar. For example:

- SOAP bindings that do not employ a URI addressing scheme may substitute another addressing scheme by replacing the `soap:address` element defined in **2.10 address Element with port [parent]** [p.19] .
- SOAP bindings that do not require a `SOAPAction` omit the `soapAction` attribute defined in **2.5 operation Element with operation [parent]** [p.12] .

2.1 Pseudo Schema (Non-Normative)

The WSDL SOAP binding described in this section extends WSDL[*WSDL 1.2 [p.29]*] by adding *element information item*s and *attribute information item*s to the Infoset of the WSDL `binding` *element information item* in the "http://www.w3.org/2003/01/wsdl" namespace. The following pseudo schema depicts each of these extensions.

```
<definitions ...>
  <binding ...>
   <soap:binding transport="uri"
                 styleDefault="document|rpc"?
                 namespaceDefault="uri"?
```

```
                encodingStyleDefault="uri"? />
  <soap:module uri="uri"
               required="boolean"? >
    <soap:propertyConstraint uri="uri"
                             type="qname"? />*
   </soap:module>*
   <operation ...>
    <soap:operation style="document|rpc"?
                    soapAction="uri"? /> ?
     <input>
      <soap:body namespace="uri"?
                 encodingStyle="uri"? /> ?
      <soap:header element="qname"?
                   type="qname"?
                   localname="nmtoken"?
                   namespace="uri"?
                   encodingStyle="uri"?
                   role="uri"? />*
      <soap:headerfault message="qname"
                        part="nmtoken"
                        namespace="uri"?
                        encodingStyle="uri"? />*
     </input>?
     <output>
      // same as input
     </output>?
     <fault>
      <soap:fault name="nmtoken"
                  namespace="uri"?
                  encodingStyle="uri"? />
     </fault>*
    </operation>*
  </binding>

  <service ...>
    <port ...>
     <soap:address location="uri" />
    </port>
  </service>
</definitions>
```

2.2 `binding` Element with `binding` [parent]

The mandatory `binding` *element information item* indicates that messages are SOAP Envelopes [*SOAP 1.2 Part 1: Messaging Framework [p.28]*].

The `binding` *element information item* has the following Infoset properties:

- A [local name] of `binding`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"

- A [parent] of `binding` in the namespace named "http://www.w3.org/2003/01/wsdl"
- One or more *attribute information item*s amongst its [attributes] as follows:
  - A required `transport` *attribute information item* as described below
  - An optional `styleDefault` *attribute information item* as described below
  - An optional `namespaceDefault` *attribute information item* as described below
  - An optional `encodingStyleDefault` *attribute information item* as described below

The `binding` *element information item* has no [children].

2.2.1 `transport` Attribute with `binding` [owner]

The `transport` *attribute information item* indicates which underlying transport to use for SOAP Envelopes within a `binding` *element information item*. The `transport` *attribute information item* has the following Infoset properties:

- A [local name] of `transport`
- A [namespace name] which has no value
- An [owner] of `binding` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `transport` *attribute information item* is *xsd:anyURI*. The value of the `transport` *attribute information item* is a URI that identifies a specific transport to carry SOAP Envelopes. The URI value "http://www.w3.org/2002/12/soap/bindings/HTTP/" corresponds to the HTTP binding in the SOAP specification.

2.2.2 `styleDefault` Attribute with `binding` [owner]

The `styleDefault` *attribute information item* indicates the default serialization style for all operations contained within the [ancestor] binding. The `styleDefault` *attribute information item* has the following Infoset properties:

- A [local name] of `styleDefault`
- A [namespace name] which has no value
- An [owner] of `binding` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `styleDefault` *attribute information item* is *xsd:string*. The value of the `styleDefault` *attribute information item* is either "document" or "rpc"; if the `styleDefault` *attribute information item* is omitted, the value is "document". See below for more information on the semantics of the `style` *attribute information item*.

2.2.3 `namespaceDefault` Attribute with `binding` [owner]

The `namespaceDefault` *attribute information item* indicates the default namespace to use for any operation contained within the [ancestor] binding with `style` *attribute information item* with value "rpc". The `namespaceDefault` *attribute information item* has the following Infoset properties:

- A [local name] of `namespaceDefault`
- A [namespace name] which has no value
- An [owner] of `binding` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `namespaceDefault` *attribute information item* is *xsd:anyURI*. The value of the `namespaceDefault` *attribute information item* is the XML namespace to use for the 'wrapper' element for the SOAP Envelope body block(s). See below for more information on the semantics of the `namespace` *attribute information item*.

2.2.4 `encodingStyleDefault` Attribute with `binding` [owner]

The `encodingStyleDefault` *attribute information item* indicates the default encodingStyle to use for any operation contained within the [ancestor] binding. The `encodingStyleDefault` *attribute information item* has the following Infoset properties:

- A [local name] of `encodingStyleDefault`
- A [namespace name] which has no value
- An [owner] of `binding` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `encodingStyleDefault` *attribute information item* is *xsd:anyURI*.

The `encodingStyle` and `encodingStyleDefault` *attribute information item*s indicate how the type description for the SOAP Envelope component was derived. This information MAY be exploited by tools that generate data structures from the type description.

The [normalized value] "http://www.w3.org/2002/12/soap-envelope/encoding/none" indicates that the type description was not derived as a function of a well-known encoding. The [normalized value] MUST NOT be "". If no `encodingStyleDefault` is in scope, then the [normalized value] is "http://www.w3.org/2002/12/soap-envelope/encoding/none"

For all [normalized value]s of the `encodingStyle` and `encodingStyleDefault` *attribute information item*s, the SOAP Envelope component(s) MUST be exactly as described by the type description; the writer of the SOAP Envelope component(s) MUST it exactly as described by the type description.

2.3 `module` Element with `binding` [parent]

The optional `module` *element information item* indicates SOAP module(s) supported by a `binding` *element information item*. A SOAP module[*SOAP 1.2 Part 1: Messaging Framework [p.28]*] realizes zero or more SOAP features implemented as one or more SOAP header blocks.

The `module` *element information item* has the following Infoset properties:

- A [local name] of `module`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"
- A [parent] of `binding` in the namespace named "http://www.w3.org/2003/01/wsdl"
- One or more *attribute information item*s amongst its [attributes] as follows:
  - A required `uri` *attribute information item* as described below
  - An optional `required` *attribute information item* as described below

2.3.1 `uri` Attribute with `module` [owner]

The `uri` *attribute information item* specifies the URI of the SOAP module for the [owner] `module` . The `uri` *attribute information item* has the following Infoset properties:

- A [local name] of `uri`
- A [namespace name] which has no value
- An [owner] of `module` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `uri` *attribute information item* is `xsd:anyURI` . The value of the `uri` *attribute information item* is a URI that identifies a specific SOAP module.

2.3.2 `required` Attribute with `binding` [owner]

The `required` *attribute information item* specifies whether the [owner] `module` is required by the [parent] binding. The `required` *attribute information item* has the following Infoset properties:

- A [local name] of `required`
- A [namespace name] which has no value
- An [owner] of `module` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `required` *attribute information item* is `xsd:boolean` .

Omitting this *attribute information item* is defined as being semantically equivalent to including it with a value of "false".

2.4 `propertyConstraint` Element with `module` [parent]

The optional `propertyConstraint` *element information item* indicates SOAP module(s) supported by a `module` *element information item*.

The `propertyConstraint` *element information item* has the following Infoset properties:

- A [local name] of `propertyConstraint`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"
- A [parent] of `module` in the namespace named "http://www.w3.org/2003/01/wsdl"
- One or more *attribute information item*s amongst its [attributes] as follows:
  - A required `uri` *attribute information item* as described below
  - An optional `type` *attribute information item* as described below

The value of the `propertyConstraint` *element information item* is the value that is required for the SOAP property of the [parent] `module` .

The `propertyConstraint` *element information item* has no [children].

2.4.1 `uri` Attribute with `propertyConstraint` [owner]

The `uri` *attribute information item* specifies the URI of the SOAP property for the [ancestor] `module` . The `uri` *attribute information item* has the following Infoset properties:

- A [local name] of `uri`
- A [namespace name] which has no value
- An [owner] of `propertyConstraint` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `uri` *attribute information item* is `xsd:anyURI` . The value of the `uri` *attribute information item* is a URI that identifies a specific SOAP property for the [ancestor] `module` .

2.4.2 `type` Attribute with `propertyConstraint` [owner]

The `type` *attribute information item* specifies the type of the value of the [owner] `propertyConstraint` . The `type` *attribute information item* has the following Infoset properties:

- A [local name] of `type`
- A [namespace name] which has no value
- An [owner] of `propertyConstraint` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `type` *attribute information item* is `xsd:QName` . The value of the `type` *attribute information item* is XML Qualified Name that identifies the type of the value for the [owner] `propertyConstraint` .

2.5 `operation` Element with `operation` [parent]

The optional `operation` *element information item* provides binding information for the [parent] operation. The `operation` *element information item* has the following Infoset properties:

- A [local name] of `operation`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"
- A [parent] of `operation` in the namespace named "http://www.w3.org/2003/01/wsdl"
- Zero or more *attribute information item*s amongst its [attributes] as follows:
  - An optional `style` *attribute information item* as described below
  - An optional `soapAction` *attribute information item* as described below

The `operation` *element information item* has no [children].

If the `operation` *element information item* has no [attributes], it MAY be omitted.

2.5.1 `style` Attribute with `operation` [owner]

The `style` *attribute information item* indicates the serialization style of the `operation` owner. The `style` *attribute information item* has the following Infoset properties:

- A [local name] of `style`
- A [namespace name] which has no value
- An [owner] of `operation` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `style` *attribute information item* is *xsd:string*.

The value of the `style` *attribute information item* indicates whether the [owner] operation is serialized according to a document-oriented style (message contains document(s)) or remote-procedure-call (RPC) oriented style (message contains parameter(s) and/or return value(s)). This information may be used to select an appropriate programming model. The value of this *attribute information item* also affects the

way in which the `Fault` of the SOAP message is constructed, as explained in section **2.6 body Element with input or output [parent]** [p.13] . If the `style` *attribute information item* is not specified, it defaults to the value specified in the [ancestor] `binding` *element information item* in the namespace named "http://www.w3.org/2003/01/wsdl/soap12".

2.5.2 `soapAction` Attribute with `operation` [owner]

Editorial note: JCS	20030115
The SOAP 1.2 HTTP binding does not define a SOAPAction HTTP header. This is a known issue.	

The `soapAction` *attribute information item* specifies the value of the HTTP SOAPAction header for the `operation` owner. The `soapAction` *attribute information item* has the following Infoset properties:

- A [local name] of `soapAction`
- A [namespace name] which has no value
- An [owner] of `operation` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `soapAction` *attribute information item* is *xsd:anyURI*.

The value of the `soapAction` *attribute information item* is the URI to be included in the HTTP SOAPAction header for the [owner] operation. This URI value should be used directly as the value for the HTTP SOAPAction header; no attempt should be made to make a relative URI value absolute when making the request. For the HTTP protocol binding of SOAP, this value is required (it has no default value). For other SOAP protocol bindings, it MUST NOT be specified.

2.6 `body` Element with `input` or `output` [parent]

The optional `body` *element information item* specifies properties of the SOAP Envelope body block(s) for the [parent] operation. The `body` *element information item* has the following Infoset properties:

- A [local name] of `body`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"
- A [parent] of `input` or `output` in the namespace named "http://www.w3.org/2003/01/wsdl"
- Zero or more *attribute information item*s amongst its [attributes] as follows:
  - An optional `namespace` *attribute information item* as described below
  - An optional `encodingStyle` *attribute information item* as described below

The `body` *element information item* has no [children].

If the `body` *element information item* has no [attributes], it MAY be omitted.

The `body` *element information item* provides information on how to assemble the different message parts inside the `Fault` element of the SOAP Envelope. The `body` *element information item* is used in both RPC-oriented and document-oriented messages, but the style of the enclosing operation has important effects on how the `Fault` section is structured:

- If the operation style is rpc each part is a parameter or a return value and appears inside a wrapper element within the body. The wrapper element is named identically to the operation name and its namespace is the value of the namespace attribute. Each message part (parameter) appears under the wrapper, represented by an accessor named identically to the corresponding parameter of the call. Parts are arranged in the same order as the parameters of the call.

- If the operation style is document there are no additional wrappers, and the message parts appear directly under the SOAP `Fault` element.

The same mechanisms are used to define the content of the `Fault` and parameter accessor elements.

Each part references a concrete schema definition using either the `element` or `type` *attribute information item*. In the first case, the element referenced by the part will appear directly under the `Body` SOAP Envelope element (for document style bindings) or under an accessor element named after the message part (in rpc style). In the second, the type referenced by the part becomes the schema type of the enclosing element (`Body` for document style or part accessor element for rpc style).

2.6.1 `namespace` Attribute with `body` [owner]

The `namespace` *attribute information item* defines the XML namespace to use for the SOAP body block(s) for the [ancestor] `operation` . The `namespace` *attribute information item* has the following Infoset properties:

- A [local name] of `namespace`

- A [namespace name] which has no value

- An [owner] of `body` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `namespace` *attribute information item* is *xsd:anyURI*. The value of the `namespace` *attribute information item* is the XML namespace to use for the 'wrapper' element for the SOAP Envelope body block(s).

If the value of the `style` *attribute information item* is "document", then the `namespace` *attribute information item* with [owner] `body` is not applicable.

2.6.2 `encodingStyle` Attribute with `body` [owner]

The `encodingStyle` *attribute information item* indicates the encoding style

that was used to generate type description for the SOAP body block(s) for the [ancestor] `operation` .

The `encodingStyle` *attribute information item* has the following Infoset properties:

- A [local name] of `encodingStyle`
- A [namespace name] which has no value
- An [owner] of `body` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `encodingStyle` *attribute information item* is *xsd:anyURI*.

The [normalized value] "http://www.w3.org/2002/12/soap-envelope/encoding/none" indicates that the type description has not been derived as a function of a well-known encoding. The [normalized value] MUST NOT be "". If no `encodingStyle` is in scope, then the [normalized value] is the `encodingStyleDefault` *attribute information item*.

2.7 `header` Element with `input` or `output` [parent]

The optional `header` *element information item* specifies SOAP header block(s) for the [ancestor] operation. The `header` *element information item* has the following Infoset properties:

- A [local name] of `header`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"
- A [parent] of `input` or `output` in the namespace named "http://www.w3.org/2003/01/wsdl"
- One or more *attribute information item*s amongst its [attributes] as follows:
  - An optional `element` *attribute information item* as described below
  - An optional `type` *attribute information item* as described below
  - An optional `localname` *attribute information item* as described below
  - An optional `namespace` *attribute information item* as described below
  - An optional `encodingStyle` *attribute information item* as described below
  - An optional `role` *attribute information item* as described below

The `header` *element information item* has no [children].

The `header` *element information item* allow headers to be defined that are transmitted as SOAP header blocks. It is not necessary to exhaustively list all header blocks that appear in the SOAP Envelope using `header` *element information item*s. For example, extensions (see Language Extensibility and Binding, [*WSDL 1.2 [p.29]*], section 4) to WSDL may imply specific headers should be added to the actual payload, and it is not required to list those headers here.

Exactly one of the `element` and `type` *attribute information item*s MUST be specified. The `localname` and `namespace` *attribute information item*s MUST be specified if and only if the `type` *attribute information item* is specified.

2.7.1 `element` Attribute with `header` [owner]

The `element` *attribute information item* refers to an XML Schema element declaration component that defines the SOAP header block for the [owner] `header` . The `element` *attribute information item* has the following Infoset properties:

- A [local name] of `element`
- A [namespace name] which has no value
- An [owner] of `header` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `element` *attribute information item* is *xsd:QName*.

2.7.2 `type` Attribute with `header` [owner]

The `type` *attribute information item* refers to an XML Schema type description that defines the SOAP header block for the [owner] `header` . The `type` *attribute information item* has the following Infoset properties:

- A [local name] of `type`
- A [namespace name] which has no value
- An [owner] of `header` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `type` *attribute information item* is *xsd:QName*.

The `namespace` *attribute information item* is used in the same way as with the `body` *element information item* (see section **2.6 body Element with input or output [parent]** [p.13]), only `style` ="document" is assumed since headers do not contain parameters.

The schema referenced MAY include definitions for the `actor` and `mustUnderstand` *attribute information item* in the namespace named "http://www.w3.org/2002/12/soap-envelope".

2.7.3 `localname` Attribute with `header` [owner]

The `localname` *attribute information item* defines the localname of the SOAP header block for the [owner] `header` when it has a `type` *attribute information item*. The `localname` *attribute information item* has the following Infoset properties:

- A [local name] of `localname`

- A [namespace name] which has no value
- An [owner] of `header` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `localname` *attribute information item* is *xsd:NCName*.

2.7.4 `namespace` Attribute with `header` [owner]

The `namespace` *attribute information item* defines the XML namespace of the SOAP header block for the [owner] `header` when it has a `type` *attribute information item*. The `namespace` *attribute information item* has the following Infoset properties:

- A [local name] of `namespace`
- A [namespace name] which has no value
- An [owner] of `header` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `namespace` *attribute information item* is *xsd:anyURI*.

2.7.5 `encodingStyle` Attribute with `header` [owner]

The `encodingStyle` *attribute information item* indicates the encoding style

that was used to generate type description for the SOAP header block(s) for the [ancestor] `operation` .

The `encodingStyle` *attribute information item* has the following Infoset properties:

- A [local name] of `encodingStyle`
- A [namespace name] which has no value
- An [owner] of `header` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `encodingStyle` *attribute information item* is *xsd:anyURI*.

The [normalized value] "http://www.w3.org/2002/12/soap-envelope/encoding/none" indicates that the type description has not been derived as a function of a well-known encoding. The [normalized value] MUST NOT be "". If no `encodingStyle` is in scope, then the [normalized value] is the `encodingStyleDefault` *attribute information item*.

2.7.6 `role` Attribute with `header` [owner]

The optional `role` *attribute information item* identifies the SOAP role to which a SOAP header block is targeted. The `role` *attribute information item* has:

- A [local name] of `role`

- A [namespace name] which has no value
- A [specified] property with a value of "true"
- An [owner] of `header` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `role` *attribute information item* is `xsd:anyURI` . The value of the `role` *attribute information item* is a URI that names a role that a SOAP node can assume. It SHOULD NOT be a relative URI.

A `role` *attribute information item* that is either absent or has an empty value is equivalent to indicating a value of "http://www.w3.org/2002/12/soap-envelope/role/ultimateReceiver", i.e., targeting the SOAP header block to an ultimate SOAP receiver.

2.8 `headerfault` Element with `input` or `output` [parent]

Editorial note: JCS	20030115
When deciding to modify header, the WG did not formally decide to modify headerfault. This subsection has not been updated pending that consideration.	

The `soap:header` and `soap:headerfault` elements allows header to be defined that are transmitted inside the `Header` element of the SOAP Envelope. It is patterned after the `soap:body` element (see section **2.6 body Element with input or output [parent]** [p.13]).

It is not necessary to exhaustively list all headers that appear in the SOAP Envelope using `soap:header` . For example, extensions (see Language Extensibility and Binding, [*WSDL 1.2 [p.29]*], section 4) to WSDL may imply specific headers should be added to the actual payload and it is not required to list those headers here.

The `namespace` attribute is used in the same way as with `soap:body` (see section **2.6 body Element with input or output [parent]** [p.13]), only `style` ="document" is assumed since headers do not contain parameters.

Together, the `message` attribute (of type `QName`) and the `part` attribute (of type `nmtoken`) reference the message part that defines the header type. The schema referenced by the part MAY include definitions for the `soap:actor` and `soap:mustUnderstand` attributes.

The referenced message need not be the same as the message that defines the SOAP `Body` .

The optional `headerfault` elements which appear inside `soap:header` and have the same syntax as `soap:header` allows specification of the header type(s) that are used to transmit error information pertaining to the header defined by the `soap:header` . The SOAP specification states that errors pertaining to headers must be returned in headers, and this mechanism allows specification of the format of such headers.

2.9 `fault` Element with `operation` [parent]

The `soap:fault` element specifies the contents of the SOAP `Fault Details` element. It is patterned after the `soap:body` element (see section **2.6 body Element with input or output [parent]** [p.13]).

The `name` attribute relates the `soap:fault` to the `wsdl:fault` defined for the operation.

The fault message MUST have a single part. The `namespace` attribute is used in the same way as with `soap:body` (see section **2.6 body Element with input or output [parent]** [p.13]), only `style` ="document" is assumed since faults do not contain parameters.

2.10 `address` Element with `port` [parent]

The mandatory `address` *element information item* contains the endpoint address of a `port` . The `address` *element information item* has the following Infoset properties:

- A [local name] of `address`
- A [namespace name] of "http://www.w3.org/2003/01/wsdl/soap12"
- A [parent] of `port` in the namespace named "http://www.w3.org/2003/01/wsdl"
- One *attribute information item* amongst its [attributes] as follows:
    - A required `location` *attribute information item* as described below

The `address` *element information item* has no [children].

2.10.1 `location` Attribute with `address` [owner]

The mandatory `location` *attribute information item* identifies the endpoint address of a `port` . The `location` *attribute information item* has the following Infoset properties:

- A [local name] of `location`
- A [namespace name] which has no value
- An [owner] of `address` in the namespace named "http://www.w3.org/2003/01/wsdl/soap12"

The type of the `location` *attribute information item* is `xsd:anyURI` . The value of the `location` *attribute information item* is a URI that processes messages bound by the `port` . The URI scheme of the value of the `location` *attribute information item* MUST correspond to the transport specified by the value of the `binding` *attribute information item* of the `port` *element information item*.

3. HTTP GET and POST Binding

WSDL includes a binding for HTTP 1.1's GET and POST [*IETF RFC 2616 [p.28]*] verbs in order to describe the interaction between a Web Browser and a web site. This allows applications other than Web Browsers to interact with the site. The following protocol specific information may be specified:

- An indication that a binding uses HTTP GET or POST
- An address for the port
- A relative address for each operation (relative to the base address defined by the port)

3.1 HTTP GET/POST Examples

The following example shows three ports that are bound differently for a given port type.

If the values being passed are "part1=1", "part2=2", "part3=3", the request format would be as follows for each port:

```
port1: GET, URL="http://example.com/o1/A1B2/3"
port2: GET, URL="http://example.com/o1?p1=1&p2=2&p3=3
port3: POST, URL="http://example.com/o1", PAYLOAD="p1=1&p2=2&p3=3"
```

For each port, the response is either a GIF or a JPEG image.

Example 1: GET and FORM POST example returning GIF or JPG

```
<definitions .... >
    <message name="m1">
        <part name="part1" type="xsd:string"/>
        <part name="part2" type="xsd:int"/>
        <part name="part3" type="xsd:string"/>
    </message>

    <message name="m2">
        <part name="image" type="xsd:binary"/>
    </message>

    <portType name="pt1">
        <operation name="o1">
           <input message="tns:m1"/>
           <output message="tns:m2"/>
        </operation>
    </portType>

    <service name="service1">
        <port name="port1" binding="tns:b1">
           <http:address location="http://example.com/"/>
        </port>
        <port name="port2" binding="tns:b2">
           <http:address location="http://example.com/"/>
        </port>
        <port name="port3" binding="tns:b3">
```

```
            <http:address location="http://example.com/"/>
        </port>
    </service>

    <binding name="b1" type="pt1">
       <http:binding verb="GET"/>
        <operation name="o1">
          <http:operation location="o1/A(part1)B(part2)/(part3)"/>
           <input>
              <http:urlReplacement/>
           </input>
           <output>
              <mime:content type="image/gif"/>
              <mime:content type="image/jpeg"/>
           </output>
        </operation>
    </binding>

    <binding name="b2" type="pt1">
       <http:binding verb="GET"/>
        <operation name="o1">
          <http:operation location="o1"/>
          <input>
              <http:urlEncoded/>
          </input>
           <output>
              <mime:content type="image/gif"/>
              <mime:content type="image/jpeg"/>
           </output>
        </operation>
    </binding>

    <binding name="b3" type="pt1">
       <http:binding verb="POST"/>
        <operation name="o1">
          <http:operation location="o1"/>
           <input>
              <mime:content type="application/x-www-form-urlencoded"/>
           </input>
           <output>
              <mime:content type="image/gif"/>
              <mime:content type="image/jpeg"/>
           </output>
        </operation>
    </binding>
</definitions>
```

3.2 How the HTTP GET/POST Binding Extends WSDL

The HTTP GET/POST Binding extends WSDL with the following extension elements:

```
<definitions .... >
    <binding .... >
       <http:binding verb="nmtoken"/>
        <operation .... >
          <http:operation location="uri"/>
```

```
            <input .... >
               <-- mime elements -->
            </input>
            <output .... >
               <-- mime elements -->
            </output>
         </operation>
    </binding>

    <port .... >
       <http:address location="uri"/>
    </port>
</definitions>
```

These elements are covered in the subsequent sections.

3.3 http:address

The `location` attribute specifies the base URI for the port. The value of the attribute is combined with the values of the `location` attribute of the `http:operation` binding element. See section **3.5 http:operation** [p.22] for more details.

3.4 http:binding

The `http:binding` element indicates that this binding uses the HTTP protocol.

```
<definitions .... >
    <binding .... >
       <http:binding verb="nmtoken"/>
    </binding>
</definitions>
```

The value of the required `verb` attribute indicates the HTTP verb. Common values are GET or POST, but others may be used. Note that HTTP verbs are case sensitive.

3.5 http:operation

The `location` attribute specifies a relative URI for the operation. This URI is combined with the URI specified in the `http:address` element to form the full URI for the HTTP request. The URI value MUST be a relative URI.

```
<definitions .... >
    <binding .... >
        <operation .... >
          <http:operation location="uri"/>
        </operation>
    </binding>
</definitions>
```

3.6 http:urlEncoded

The `urlEncoded` element indicates that all the message parts are encoded into the HTTP request URI using the standard URI- encoding rules ("name1=value&name2=value "). The names of the parameters correspond to the names of the message parts. Each value contributed by the part is encoded using a "name=value" pair. This may be used with GET to specify URL encoding, or with POST to specify a FORM-POST. For GET, the "?" character is automatically appended as necessary.

```
<http:urlEncoded/>
```

For more information on the rules for URI-encoding parameters, see Form submission ([*HTML 4.01 [p.27]*], section 17.13), Ampersands in URI ([*HTML 4.01 [p.27]*], section B.2.2), and Form content types ([*HTML 4.01 [p.27]*], section 17.13.4).

3.7 http:urlReplacement

The `http:urlReplacement` element indicates that all the message parts are encoded into the HTTP request URI using a replacement algorithm:

- The relative URI value of `http:operation` is searched for a set of search patterns.
- The search occurs before the value of the `http:operation` is combined with the value of the location attribute from `http:address` .
- There is one search pattern for each message part. The search pattern string is the name of the message part surrounded with parenthesis "(" and ")".
- For each match, the value of the corresponding message part is substituted for the match at the location of the match.
- Matches are performed before any values are replaced (replaced values do not trigger additional matches).

Message parts MUST NOT have repeating values.

```
<http:urlReplacement/>
```

4. MIME Binding

WSDL includes a way to bind abstract types to concrete messages in some MIME format. Bindings for the following MIME types are defined:

- "multipart/related", defined in [*IETF RFC 2387 [p.28]*].
- "text/xml", defined in [*IETF RFC 3023 [p.28]*].

- "application/x-www-form-urlencoded", defined in Form content types ([*HTML 4.01 [p.27]*], section 17.13.4).
- Others (by specifying the MIME type string)

The set of defined MIME types is both large and evolving, so it is not a goal for WSDL to exhaustively define XML grammar for each MIME type. Nothing precludes additional grammar to be added to define additional MIME types as necessary. If a MIME type string is sufficient to describe the content, the `mime` element defined below can be used.

4.1 MIME Binding example

Editorial note: JJM	20020301
The following examples are SOAP 1.1 examples, not SOAP 1.2 examples.	

This example describes that a GetCompanyInfo SOAP request may be sent to a StockQuote service via the SOAP HTTP binding. The request takes a ticker symbol of type string. The response contains multiple parts encoded in the MIME format multipart/related: a SOAP Envelope containing the current stock price as a float, zero or more marketing literature documents in HTML format, and an optional company logo in either GIF or JPEG format.

Example 2: Using multipart/related with SOAP

```
<definitions .... >

    <types>
        <schema .... >
           <element name="GetCompanyInfo">
               <complexType>
                   <all>
                       <element name="tickerSymbol " type="string"/>
                   </all>
               </complexType>
           </element>
           <element name="GetCompanyInfoResult">
               <complexType>
                   <all>
                       <element name="result" type="float"/>
                   </all>
               </complexType>
           </element>
           <complexType name="ArrayOfBinary">
               <complexContent>
                   <restriction base="soapenc:Array">
                      <attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:binary[]"/>
                   </restriction>
               <complexContent>
           </complexType>
        </schema>
    </types>
```

```
    <message name="m1">
        <part name="body" element="tns:GetCompanyInfo"/>
    </message>

    <message name="m2">
        <part name="body" element="tns:GetCompanyInfoResult"/>
        <part name="docs" type="xsd:string"/>
        <part name="logo" type="tns:ArrayOfBinary"/>
    </message>

    <portType name="pt1">
        <operation name="GetCompanyInfo">
           <input message="m1"/>
           <output message="m2"/>
        </operation>
    </portType>

    <binding name="b1" type="tns:pt1">
        <operation name="GetCompanyInfo">
          <soap:operation soapAction="http://example.com/GetCompanyInfo"/>
           <input>
              <soap:body use="literal"/>
           </input>
           <output>
              <mime:multipartRelated>
                  <mime:part>
                      <soap:body parts="body" use="literal"/>
                  </mime:part>
                  <mime:part>
                      <mime:content part="docs" type="text/html"/>
                  </mime:part>
                  <mime:part>
                      <mime:content part="logo" type="image/gif"/>
                      <mime:content part="logo" type="image/jpeg"/>
                  </mime:part>
              </mime:multipartRelated>
           </output>
        </operation>
    </binding>

    <service name="CompanyInfoService">
        <port name="CompanyInfoPort"binding="tns:b1">
          <soap:address location="http://example.com/companyinfo"/>
        </port>
    </service>
</definitions>
```

4.2 How the MIME Binding extends WSDL

The MIME Binding extends WSDL with the following extension elements:

```
<mime:content part="nmtoken"? type="string"?/>

<mime:multipartRelated>
    <mime:part> *
        <-- mime element -->
    </mime:part>
</mime:multipartRelated>
<mime:mimeXml part="nmtoken"?/>
```

They are used at the following locations in WSDL:

```
<definitions .... >
    <binding .... >
        <operation .... >
           <input .... >
              <-- mime elements -->
           </input>
           <output .... >
              <-- mime elements -->
           </output>
        </operation>
    </binding>
</definitions>
```

MIME elements appear under input and output to specify the MIME format. If multiple appear, they are considered to be alternatives.

4.3 mime:content

To avoid having to define a new element for every MIME format, the `mime:content` element may be used if there is no additional information to convey about the format other than its MIME type string.

```
<mime:content part="nmtoken"? type="string"?/>
```

The `part` attribute is used to specify the name of the message part. If the message has a single part, then the `part` attribute is optional. The `type` attribute contains the MIME type string. A `type` value has two portions, separated by a slash (/), either of which may be a wildcard (*). Not specifying the `type` attribute indicates that all MIME types are acceptable.

If the return format is XML [*XML 1.0 [p.28]*], but the schema is not known ahead of time, the generic mime element can be used indicating "text/xml" [*IETF RFC 3023 [p.28]*]:

```
<mime:content type="text/xml"/>
```

A wildcard (*) can be used to specify a family of mime types, for example all text types.

```
<mime:content type="text/*"/>
```

The following two examples both specify all mime types:

```
<mime:content type="*/*"/>
<mime:content/>
```

4.4 mime:multipartRelated

The "multipart/related" MIME type aggregates an arbitrary set of MIME formatted parts into one message using the MIME type "multipart/related". The `mime:multipartRelated` element describes the concrete format of such a message:

```
<mime:multipartRelated>
    <mime:part> *
        <-- mime element -->
    </mime:part>
</mime:multipartRelated>
```

The `mime:part` element describes each part of a "multipart/related" message [*IETF RFC 2387 [p.28]*]. MIME elements appear within `mime:part` to specify the concrete MIME type for the part. If more than one MIME element appears inside a `mime:part` , they are alternatives.

4.5 soap:body

When using the MIME binding with SOAP requests [*SOAP 1.2 Part 1: Messaging Framework [p.28]*], it is legal to use the `soap:body` element as a MIME element. It indicates the content type is "text/xml", and there is an enclosing SOAP Envelope.

4.6 mime:mimeXml

To specify XML payloads that are not SOAP compliant (do not have a SOAP Envelope), but do have a particular schema, the `mime:mimeXml` element may be used to specify that concrete schema. The `part` attribute refers to a message part defining the concrete schema of the root XML element. The `part` attribute MAY be omitted if the message has only a single part. The part references a concrete schema using the `element` attribute for simple parts or `type` attribute for composite parts .

```
<mime:mimeXml part="nmtoken"?/>
```

5. References

5.1 Normative References

[HTML 4.01]
HTML 4.01 Specification, D. Raggett, A. Le Hors, I. Jacobs, Editors. World Wide Web Consortium, 24 December 1999. This version of the HTML 4.01 Recommendation is http://www.w3.org/TR/1999/REC-html401-19991224. The latest version of HTML 4.01 is available at http://www.w3.org/TR/html401.

[IETF RFC 2045]
Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, N. Freed, N. Borenstein, Authors. Internet Engineering Task Force, November 1996. Available at

http://www.ietf.org/rfc/rfc2045.txt.

[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.

[IETF RFC 2387]
The MIME Multipart/Related Content-type, E. Levinson, Authors. Internet Engineering Task Force, August 1998. Available at http://www.ietf.org/rfc/rfc2387.txt.

[IETF RFC 2396]
Uniform Resource Identifiers (URI): Generic Syntax, T. Berners-Lee, R. Fielding, L. Masinter, Authors. Internet Engineering Task Force, August 1998. Available at http://www.ietf.org/rfc/rfc2396.txt.

[IETF RFC 2616]
Hypertext Transfer Protocol -- HTTP/1.1, R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, Authors. Internet Engineering Task Force, June 1999. Available at http://www.ietf.org/rfc/rfc2616.txt.

[IETF RFC 3023]
XML Media Types, M. Murata, S. St. Laurent, D. Kohn, Authors. Internet Engineering Task Force, January 2001. Available at http://www.ietf.org/rfc/rfc3023.txt.

[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, 19 December 2002. This version of the SOAP Version 1.2 Part 1 Specification is http://www.w3.org/TR/2002/CR-soap12-part1-20021219/. The latest version of SOAP Version 1.2 Part 1 is available at http://www.w3.org/TR/soap12-part1/.

[SOAP 1.2 Part 2: Adjuncts]
SOAP Version 1.2 Part 2: Adjuncts, M. Gudgin, M. Hadley, N. Mendelsohn, J-J. Moreau, and H. Frystyk Nielsen, Editors. World Wide Web Consortium, 19 December 2002. This version of the SOAP Version 1.2 Part 2 Specification is http://www.w3.org/TR/2002/CR-soap12-part2-20021219/. The latest version of SOAP Version 1.2 Part 2 is available at http://www.w3.org/TR/soap12-part2/.

[XML 1.0]
Extensible Markup Language (XML) 1.0 (Second Edition), T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler, Editors. World Wide Web Consortium, 10 February 1998, revised 6 October 2000. This version of the XML 1.0 Recommendation is http://www.w3.org/TR/2000/REC-xml-20001006. The latest version of XML 1.0 is available at http://www.w3.org/TR/REC-xml.

[XML Information Set]
XML Information Set, J. Cowan and R. Tobin, Editors. World Wide Web Consortium, 24 October 2001. This version of the XML Information Set Recommendation is http://www.w3.org/TR/2001/REC-xml-infoset-20011024. 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, H. Thompson, D. Beech, M. Maloney, and N. Mendelsohn, Editors. World Wide Web Consortium, 2 May 2001. This version of the XML Schema Part 1 Recommendation is http://www.w3.org/TR/2001/REC-xmlschema-1-20010502. 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, P. Byron and A. Malhotra, Editors. World Wide Web Consortium, 2 May 2001. This version of the XML Schema Part 2 Recommendation is http://www.w3.org/TR/2001/REC-xmlschema-2-20010502. The latest version of XML Schema Part 2 is available at http://www.w3.org/TR/xmlschema-2.

[WSDL 1.2]
Web Services Description Language (WSDL) Version 1.2, R. Chinnici, M. Gudgin, J-J. Moreau, S. Weerawarana, Editors. World Wide Web Consortium, 24 January 2003. This version of the Web Services Description Version 1.2 Specification is available is available at http://www.w3.org/TR/2003/WD-wsdl12-20030124. The latest version of Web Services Description Version 1.2 is available at http://www.w3.org/TR/wsdl12.

5.2 Informative References

[WSDL 1.2 Primer]
Web Services Description (WSDL) Version 1.2: Primer, K. Sankar, K. Liu, D. Booth, Editors. World Wide Web Consortium, 24 January 2003. The editors' version of the Web Services Description Version 1.2: Primer document is available from http://www.w3.org/2002/ws/desc/.

A. Acknowledgements (Non-Normative)

This document is the work of the W3C Web Services Description Working Group.

Members of the Working Group are (at the time of writing, and by alphabetical order): Adi Sakala (IONA Technologies), Allen Brookes (Rogue Wave Softwave), Amelia Lewis (TIBCO/Extensibility), Arthur Ryman (IBM), Barbara Zengler (DaimlerChrysler Research and Technology), Dale Moberg (Cyclone Commerce), Dan Kulp (IONA Technologies), Daniel Schutzer (Citigroup), Dave Solo (Citigroup), David Booth (W3C), Dietmar Gaertner (Software AG), Don Mullen (TIBCO Software), Erik Ackerman (Lexmark), Glen Daniels (Macromedia), Igor Sedukhin (Computer Associates), Jacek Kopecky (Systinet), Jean-Jacques Moreau (Canon), Jeff Mischkinsky (Oracle Corporation), Jeffrey Schlimmer (Microsoft Corporation), Jerry Thrasher (Lexmark), Johan Pauhlsson (L'Échangeur), Jonathan Marsh (*Chair*, Microsoft Corporation), Kevin Canyang Liu (SAP), Laurent De Teneuille (L'Échangeur), Lily Liu (webMethods, Inc.), Martin Gudgin (Microsoft Corporation), Michael Champion (Software AG), Michael Mahan (Nokia), Michael Mealling (Verisign), Mike Ballantyne (Electronic Data Systems), Mike Davoren (W. W. Grainger), Mike McHugh (W. W. Grainger), Philippe Le Hégaret (W3C), Prasad Yendluri (webMethods, Inc.), Roberto Chinnici (Sun Microsystems), Sandeep Kumar (Cisco Systems), Sandra Swearingen (U.S. Department of Defense, U.S. Air Force), Sanjiva Weerawarana (IBM), Stefano Pogliani (Sun Microsystems), Stephen White (SeeBeyond), Steve Graham (Global Grid Forum), Steve Lind (AT&T), Steve Tuecke (Global Grid Forum), Tim Finin (University of Maryland), Tom Jordahl (Macromedia), Umit Yalcinalp (Oracle Corporation), Waqar Sadiq (Electronic Data Systems), William Stumbo (Xerox), William Vambenepe (Hewlett-Packard Company), Youenn Fablet (Canon)

Previous members were: Aaron Skonnard (DevelopMentor), Don Wright (Lexmark), Jochen Ruetschlin (DaimlerChrysler Research and Technology), Joyce Yang (Oracle Corporation), Keith Ballinger (Microsoft), Krishna Sankar (Cisco Systems), Mario Jeckle (DaimlerChrysler Research and Technology), Pallavi Malu (Intel Corporation)

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

B. Part 2 Change Log (Non-Normative)

B.1 WSDL Specification Changes

Date	Author	Description
20030117	JCS	Incorporated resolution for Issue 5 (@encodingStyle). Referenced (rather than in-lined XML Schema).
20030117	JJM	Various editorial fixes.
20030116	JCS	Updated pseudo and XML Schema.
20030116	JJM	Added propertyConstraint section.
20030116	JJM	Added soap:module section.
20030115	JCS	Incorporated resolutions for Issue 25 (drop @use and @encoding), Issue 51 (headers reference element/type), and attribute rollup into text and schema. Began reworking SOAP HTTP binding to use Infoset model. Removed informative appendices 'Notes on URIs' and example WSDL documents; expect them to appear in the primer. Updated SOAP 1.2 references to CR.
20030114	JJM	Removed ednote saying Part 2 is out of synch with Part 1.
20030111	JJM	Incorporated resolution for issue 17 (role AII).
20030109	JJM	Incorporated resolution for issue 4 (Namespaces).
20020702	JJM	Added summary to prefix table.
20020628	JJM	Added out-of-synch-with-Part2 and not-soap12-yet ednote.
20020621	JJM	Commented out the link to the previous version. There is no previous version for 1.2 right now.
20020621	JJM	Rewrote the Notation Conventions section.
20020621	JJM	Added reference to part 0 in introduction. Renumbered references.
20020621	JJM	Simplified abstract and introduction.
20020621	JJM	Obtain the list of WG members from a separate file.
20020621	JJM	Updated stylesheet and DTDs to latest XMLP stylesheet and DTDs.
20020621	JJM	Deleted placeholder for appendix C "Location of Extensibility Elements", since this is part 1 stuff and extensibility has been reworked anyway.

20020621	JJM	Corrected link to issues lists
20020621	JJM	Updated title from "WSDL" to "Web Services Description Language". Now refer to part 1 as "Web Services... Part 1: Framework
20020621	JJM	Added Jeffrey as an editor :-). Removed Gudge (now on Part 2) :-(
20020411	JJM	Fixed typos noticed by Kevin Liu
20020301	JJM	Converted the "Schemas" sections
20020301	JJM	Converted the "Wire WSDL examples" sections
20020301	JJM	Converted the "Notes on URIs" sections
20020301	JJM	Converted the "Notational Conventions" sections
20020301	JJM	Converted the "References" sections
20020301	JJM	Converted the "MIME Binding" section to XML
20020221	JJM	Converted the "HTTP Binding" section to XML
20020221	JJM	Added placeholders for the "Wire examples" and "Schema" sections
20020221	JJM	Converted the "SOAP Binding" section to XML
20020221	JJM	Added the Change Log
20020221	JJM	Added the Status section
20020221	JJM	Simplified the introduction; referred to Part1 for a longer introduction
20020221	JJM	Renamed to "Part 2: Bindings"
20020221	JJM	Created from http://www.w3.org/TR/2001/NOTE-wsdl-20010315

B.2 XML Schema Changes

The encoding schema has been updated to be compliant with the XML Schema Recommendations ([*XML Schema Structures [p.28]*] and [*XML Schema Datatypes [p.29]*]). The table below shows the categories of change.

Class	Meaning
@@@	@@@

The table below lists the changes to the encoding schema.

Class	Description
@@@	@@@