



# What's New in WSDL 2.0

---

Arthur Ryman  
IBM Rational Software  
[ryman@ca.ibm.com](mailto:ryman@ca.ibm.com)





# Abstract

---

The W3C Web Services Description Working Group is currently developing the Web Services Description Language (WSDL) 2.0 Specification. WSDL 2.0 has many new features such as interface inheritance, extensible message exchange patterns, and an abstract component model. This talk gives an overview of WSDL 2.0, explains how it differs from WSDL 1.1, and discusses Open Source implementation plans in the Eclipse Web Tools Platform Project and the Apache Woden Project.

This is an Intermediate Level talk. It assumes familiarity with Web Services, XML Schema 1.0 and WSDL 1.1.



# My Background

---

- Software Development Manager at IBM Toronto Lab
  - Focus on Web Service, XML, and J2EE Tools
  - Rational Application Developer V6
  - WebSphere Studio Application Developer V4, V5
  - VisualAge for Java V1, V2, V3
- Leader of Web Standard Tools subproject, Eclipse Web Tools Platform project
- Editor of W3C WSDL 2.0 Core Language specification
- Committer on Apache Woden Project



# Topics

---

- Introduction
- Overview
- Component Model
- Call for Participation



# Introduction

---





# The Role of WSDL

---

- Web Services have emerged as the industry standard for application integration over the Internet (and the intranet)
- WSDL is a central development artifact
- WSDL defines the contract between a service and its clients
- WSDL drives code generation of client proxies and server skeletons
- WSDL gets generated, edited, validated, tested, and viewed

# W3C Web Service Description Working Group

---

- Formed to evolve the WSDL 1.1 Note into a W3C Recommendation
- Support for SOAP 1.2
- Incorporate clarifications and best practices as reflected in the WS-I Basic Profile 1.0
- Provide an extensible basis so new requirements can be supported through other specifications
- Originally called WSDL 1.2, but later renamed to WSDL 2.0 due to the many changes from WSDL 1.1



# Deliverables

---

- Available at <http://www.w3.org/2002/ws/desc/>
- Requirements
- Usage Scenarios
- Part 0 – Primer
- Part 1 – Core Language
  - XML Schema
- Part 2 – Adjuncts
  - XML Schema for SOAP 1.2 Binding
  - XML Schema for HTTP Binding
- Assigning Media Types to Binary Data in XML
- SOAP 1.1 Binding Note
- RDF Mapping
- Test Suite



# Part 1 – Core Language

---

- The core language is described in terms of an abstract Component Model
- The concrete language is specified as an XML Infoset and has a normative XML schema
- The mapping from the XML Infoset to the Component Model is specified
- The English description is complemented with a non-normative formal specification language, Z Notation, for use by implementers



## Part 2 – Adjuncts

---

- Message Exchange Patterns
  - In-Only, Robust-In-Only
  - Out-Only, Robust-Out-Only
  - In-Out, In-Optional-Out
  - Out-In, Out-Optional-In
- Operation Styles
  - RPC
  - IRI
  - Multipart
- Operation Safety

# Part 2 – Adjuncts – Binding Extensions

---

- SOAP 1.2 Binding
  - Any protocol
  - More detail for HTTP
- HTTP Binding
  - Motivated by REST requirements



# Test Suite

---

- Publicly available in W3C CVS repository:
  - <http://dev.w3.org/cvsweb/2002/ws/desc/test-suite/>
- Standard test bucket for use by implementation
- Helpful source of examples
- All WSDL 2.0 documents in test suite are XML Schema valid
- Provides good and bad WSDL 2.0 documents
- Coverage goals:
  - Good test cases cover all elements and attributes
  - Bad test cases cover all error conditions
- If you have a question about the specification, write a test case and contribute it to [www-ws-desc@w3.org](mailto:www-ws-desc@w3.org)



# New and Noteworthy

---

- Some name changes:
  - Old: <portType>, <port>, <definitions>
  - New: <interface>, <endpoint>, <description>
- An <interface> may extend one or more other <interface>
  - Motivated by Grid
- No more <message>/<part> construct
  - Messages are abstractly described by XML schema
- A <service> implements an <interface>
  - Each <endpoint> of the <service> binds the same <interface>
- Use of <service> element for transmitting Web service URLs
- Document composition *via* <import> and <include>
- Abstract Component Model



# New and Noteworthy *(Continued)*

---

- `<fault>` has been promoted to be a sibling of `<operation>`
  - Enables reuse of faults across operations and via interface extension
- SOAP Encoding is deprecated
  - Messages must have accurate XML schemas
- New extensibility mechanisms
  - Features and Properties
  - Message Exchange Patterns
- New IANA media type: `application/wsdl+xml`
  - New fragment syntax for components
- URI References for WSDL components
  - Enables other specifications to reference parts of a WSDL document



# Extensibility

---

- WSDL 2.0 has an open content model
  - Other specifications may add elements and attributes from other namespaces
  - *e.g.* this is how bindings are handled
- XML Schema is the favored type system
  - Others, *e.g.* DTD, RELAXNG, are allowed
- <feature> and <property> allows required protocol features to be specified and values set for them
  - *e.g.* security
- Other specifications may extend the component model
- Other specifications may define new message exchange patterns



# Example WSDL 2.0 Document

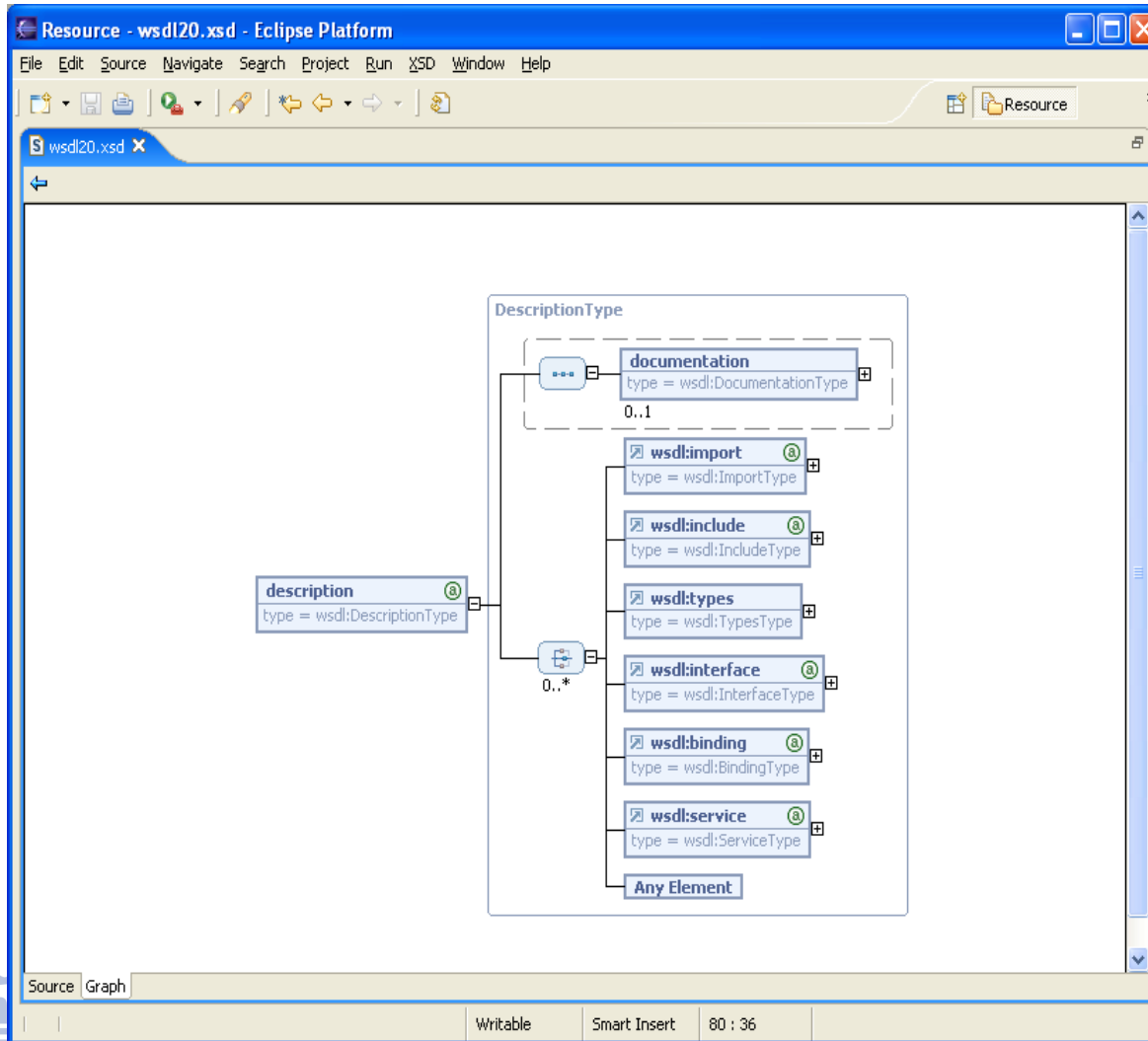
---

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:description
  targetNamespace="http://example.org/TicketAgent.wsdl20"
  xmlns:xsTicketAgent="http://example.org/TicketAgent.xsd"
  xmlns:wsdl="http://www.w3.org/@@@@/@@/wsdl"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <xs:import schemaLocation="TicketAgent.xsd"
      namespace="http://example.org/TicketAgent.xsd" />
  </wsdl:types>

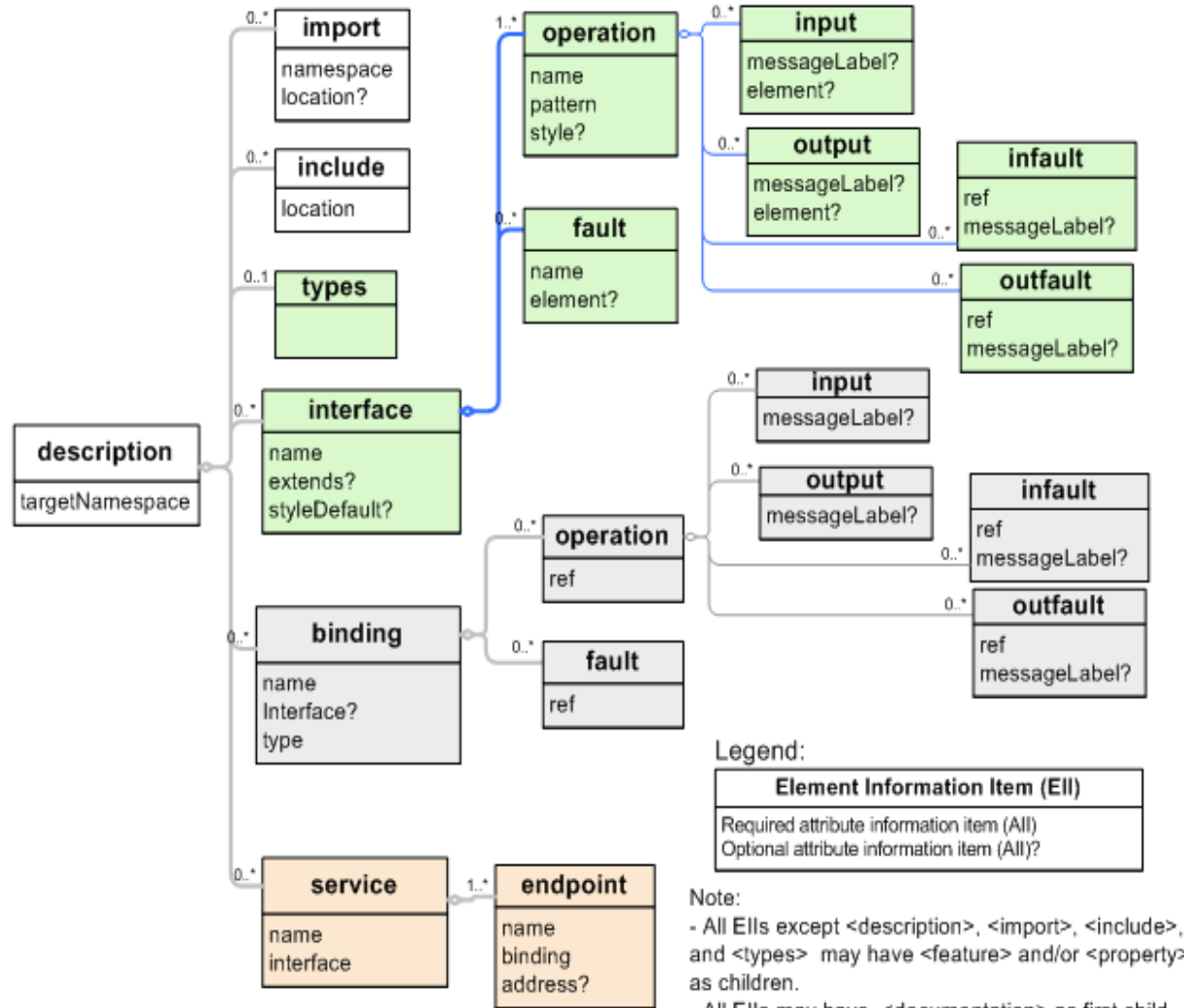
  <wsdl:interface name="TicketAgent">
    <wsdl:operation name="listFlights"
      pattern="http://www.w3.org/@@@@/@@/wsdl/in-out">
      <wsdl:input element="xsTicketAgent:listFlightsRequest"/>
      <wsdl:output element="xsTicketAgent:listFlightsResponse"/>
    </wsdl:operation>

    <wsdl:operation name="reserveFlight"
      pattern="http://www.w3.org/@@@@/@@/wsdl/in-out">
      <wsdl:input element="xsTicketAgent:reserveFlightRequest"/>
      <wsdl:output element="xsTicketAgent:reserveFlightResponse"/>
    </wsdl:operation>
  </wsdl:interface>
</wsdl:description>
```

# WSDL 2.0 XSD



# WSDL 2.0 XML Infoset Model





# <import> and <include>

---

- WSDL 2.0 documents can be modularized and composed
- Modeled on <xs:import> and <xs:include> from XML Schema
- <include> brings in components from the same namespace
  - Required @location attribute (xs:anyURI)
  - Is transitive
- <import> brings in components from other namespace
  - Required @namespace attribute (xs:anyURI)
  - Optional @location attribute (xs:anyURI)
  - Is non-transitive



## <types>

---

- Defines types used by messages
- XML Schema is primary type system
  - <xs:schema> defines schema components inline
  - <xs:import> refers to schema components defined elsewhere
- Other XML type systems MAY be used, *e.g.*
  - DTD
  - RELAXNG
  - Schematron
- Non-XML type systems are allowed



# <interface>

---

- Defines <faults> and <operations>
- <operation> supports a Message Exchange Pattern (e.g. Request-Response)
  - Contains zero or more <input>, <output>, <infault>, <outfault>
- An interface MAY extend zero or more other interfaces
  - Diamond inheritance allowed
  - No operator overriding allowed



## <binding>

---

- Associates a concrete protocol and format with an operation
- Structural similar to <operation> but includes binding details via extension elements and attributes
- Bindings for SOAP 1.2 and HTTP are defined
- Binding for SOAP 1.1 will be a W3C Note
- Default bindings can be defined
  - Independent of any interface
  - Reusable in service endpoints



## <service>

---

- Defines a set of Web service endpoints
- Implements a single <interface>
- Each <endpoint> supports a <binding> for the same interface
  - Optional @address attribute (xs:anyURI)
- ServiceType complex type MAY be used to transmit Web service references in messages
  - Optionally <xs:restrict> ServiceType to have a fixed value for @interface
- Global wdsli:wsdLocation attribute MAY be used to provide the location of a WSDL 2.0 namespace
  - Modelled on global xsi:schemaLocation attribute



# Component Model

---





# Component Model

---

- WSDL 2.0 is specified in terms of an abstract component model
- Components have properties which may be references to other components
- Similar to XML Infoset and XML Schema component models
- Like an object model, but just models state
- Uses mathematical structures such as sets
- Simplifies statement of constraints
- Improves precision
- Combines multiple WSDL and XSD documents
- Maps to XML syntax



# Component Types

---

- Container
  - Description
- Types
  - Element Declaration, Type Definition
- Interfaces
  - Interface, Interface Fault, Interface Operation, Interface Message Reference, Interface Fault Reference
- Bindings
  - Binding, Binding Fault, Binding Operation, Binding Message Reference, Binding Fault Reference
- Services
  - Service, Endpoint
- Features and Properties
  - Feature, Property
- Extensions
  - Typically binding extensions, *e.g.* SOAP 1.2, HTTP



# Description

---

- Container for type system components and top-level WSDL components
- Maps to <description>
- **{interfaces}** OPTIONAL. A set of Interface components.
- **{bindings}** OPTIONAL. A set of Binding components.
- **{services}** OPTIONAL. A set of Service components.
- **{element declarations}** OPTIONAL. A set of Element Declaration components.
- **{type definitions}** OPTIONAL. A set of Type Definition components.



# Interface

---

- Describes the operations of a service
- Maps to <interface>
- **{name}** REQUIRED. An *xs:QName*.
- **{extended interfaces}** OPTIONAL. A set of declared [Interface](#) components which this interface extends.
- **{interface faults}** OPTIONAL. The set of declared [Interface Fault](#) components. The namespace name of the [name](#) property of each [Interface Fault](#) in this set MUST be the same as the namespace name of the [name](#) property of this [Interface](#) component.
- **{interface operations}** OPTIONAL. A set of declared [Interface Operation](#) components. The namespace name of the [name](#) property of each [Interface Operation](#) in this set MUST be the same as the namespace name of the [name](#) property of this [Interface](#) component.



# Interface Fault

---

- Describes the faults that may be thrown by the service.
- Maps to <interface><fault>
- **{name}** REQUIRED. An *xs:QName*.
- **{element declaration}** OPTIONAL. A reference to a Element Declaration component in the {element declarations} property of the Description component. This element represents the content or “payload” of the fault.

# Interface Operation

- Describes the operations of a service
- Maps to <interface><operation>
- **{name}** REQUIRED. An *xs:QName*.
- **{message exchange pattern}** REQUIRED. An *xs:anyURI* identifying the message exchange pattern used by the operation. This *xs:anyURI* MUST be an absolute IRI (see [[IETF RFC 3987](#)]).
- **{interface message references}** OPTIONAL. A set of [Interface Message Reference](#) components for the ordinary messages the operation accepts or sends.
- **{interface fault references}** OPTIONAL. A set of [Interface Fault Reference](#) components for the fault messages the operation accepts or sends.
- **{style}** OPTIONAL. A set of *xs:anyURIs* identifying the rules that were used to construct the **{element declaration}** properties of **{interface message references}**. (See [2.4.1.1 Operation Style](#).) These *xs:anyURIs* MUST be absolute IRIs (see [[IETF RFC 3986](#)]).

# Interface Message Reference

- Describes the direction and format of a message in an operation
- Maps to <interface><operation>(<input>|<output>)
- **{message label}** REQUIRED. An *xs:NCName*. This property identifies the role this message plays in the **{message exchange pattern}** of the **Interface Operation** component this message is contained within. The value of this property MUST match the name of a placeholder message defined by the message exchange pattern.
- **{direction}** REQUIRED. An *xs:token* with one of the values *in* or *out*, indicating whether the message is coming to the service or going from the service, respectively. The direction MUST be the same as the direction of the message identified by the **{message label}** property in the **{message exchange pattern}** of the **Interface Operation** component this is contained within.
- **{message content model}** REQUIRED. An *xs:token* with one of the values *#any*, *#none*, *#other*, or *#element*. **{element declaration}** OPTIONAL. A reference to an XML element declaration in the **{element declarations}** property of the Description component. This element represents the content or “payload” of the message.

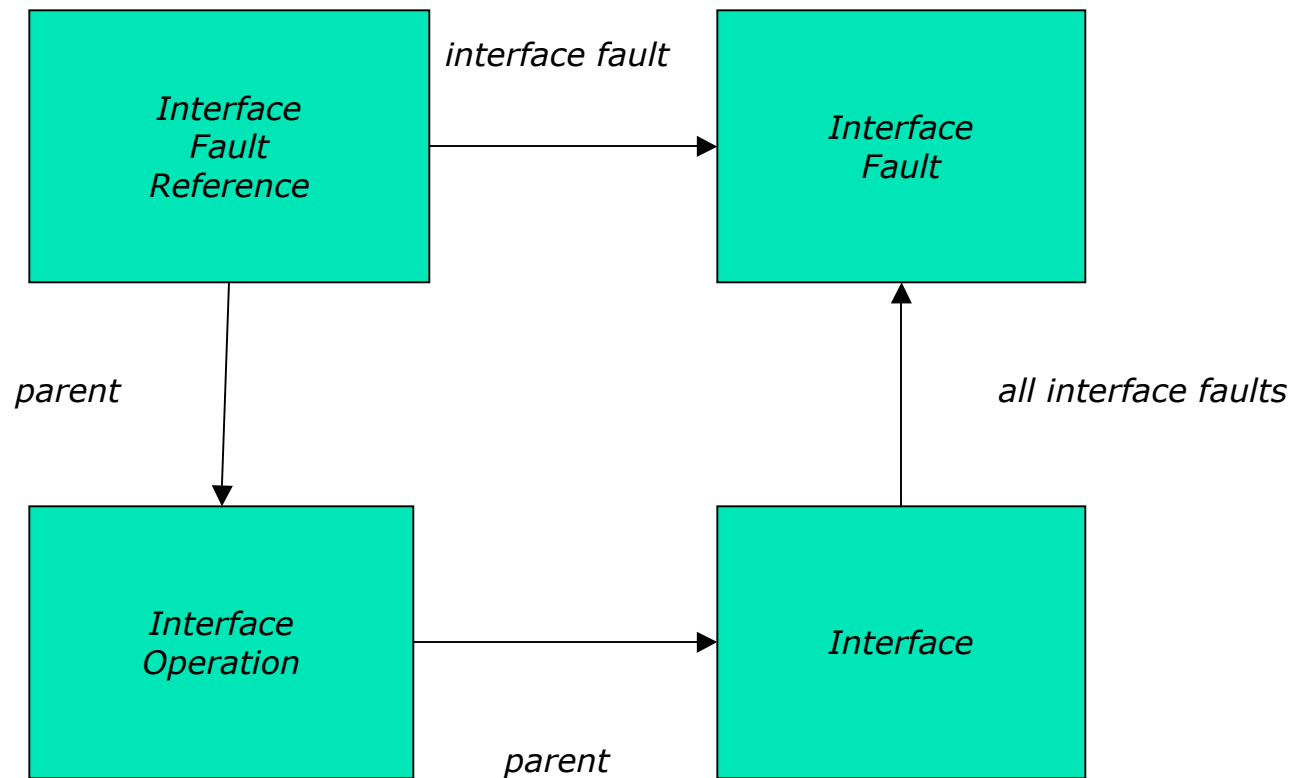


# Interface Fault Reference

---

- Describes a fault thrown by an operation
- Maps to <interface><operation>(<infault>|<outfault>)
- **{interface fault}** REQUIRED. An Interface Fault component in the **{interface faults}** property of the [parent] Interface Operation component's [parent] Interface component, or an Interface component that it directly or indirectly extends.
- **{message label}** REQUIRED. An *xs:NCName*. This property identifies the message this fault relates to among those defined in the **{message exchange pattern}** property of the Interface Operation component it is contained within.
- **{direction}** REQUIRED. A *xs:token* with one of the values *in* or *out*, indicating whether the fault is coming to the service or going from the service, respectively.

# Interface Fault Reference Consistency





# Binding

---

- Describes the concrete protocol and format of the messages.
- Maps to <wsdl:binding>
- **{name}** REQUIRED. An *xs:QName*.
- **{interface}** OPTIONAL. An [Interface](#) component indicating the interface for which binding information is being specified.
- **{type}** REQUIRED. An *xs:anyURI*. This *xs:anyURI* MUST be an absolute IRI as defined by [[IETF RFC 3987](#)]. The value of this IRI indicates what kind of concrete binding details are contained within this [Binding](#) component. Specifications (such as [[WSDL 2.0 Adjuncts](#)]) that define such concrete binding details MUST specify appropriate values for this property. The value of this property MAY be the namespace name of the extension elements or attributes which define those concrete binding details.
- **{binding faults}** OPTIONAL. A set of [Binding Fault](#) components.
- **{binding operations}** OPTIONAL. A set of [Binding Operation](#) components.

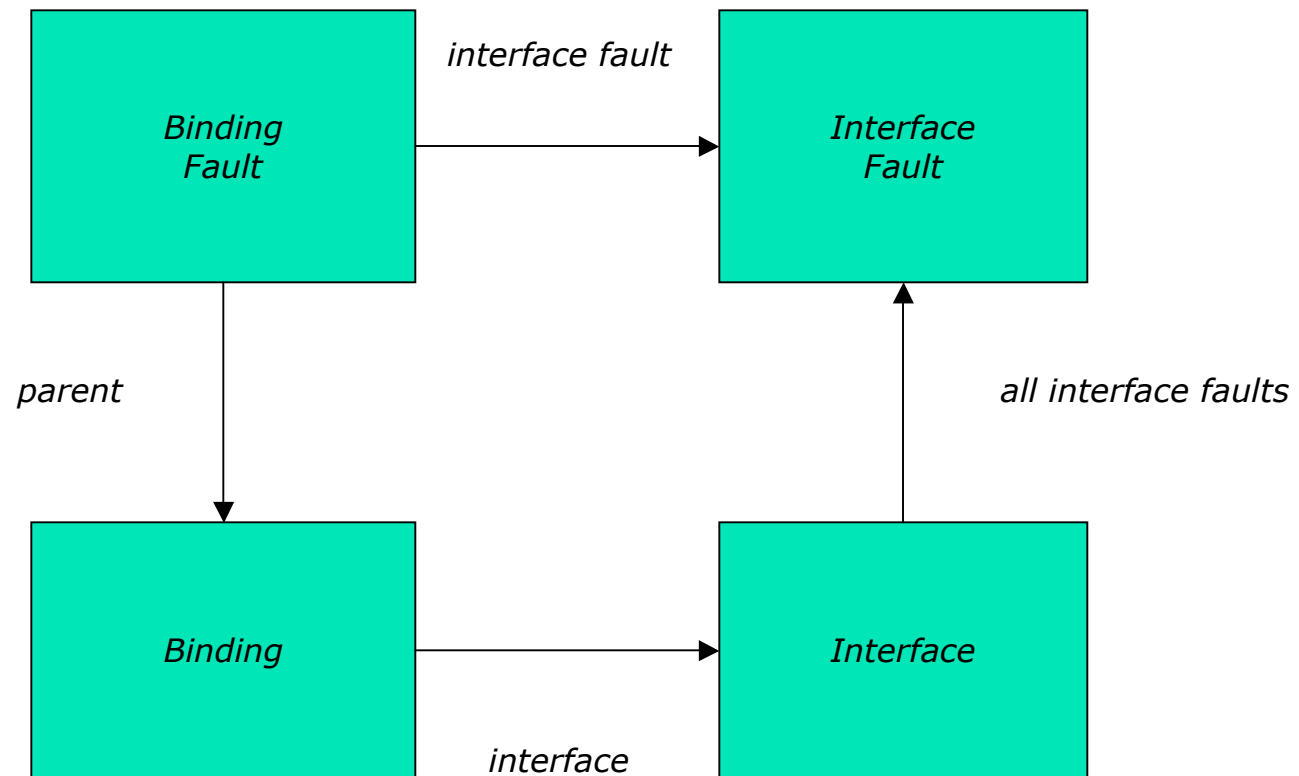


# Binding Fault

---

- Describes binding details for a fault that may be thrown by a service
- Maps to `<binding><fault>`
- **{interface fault}** REQUIRED. An Interface Fault component in the {interface faults} property of the Interface component identified by the {interface} property of the parent Binding component, or an Interface component that that Interface component directly or indirectly extends. This is the Interface Fault component for which binding information is being specified.

# Binding Fault Consistency



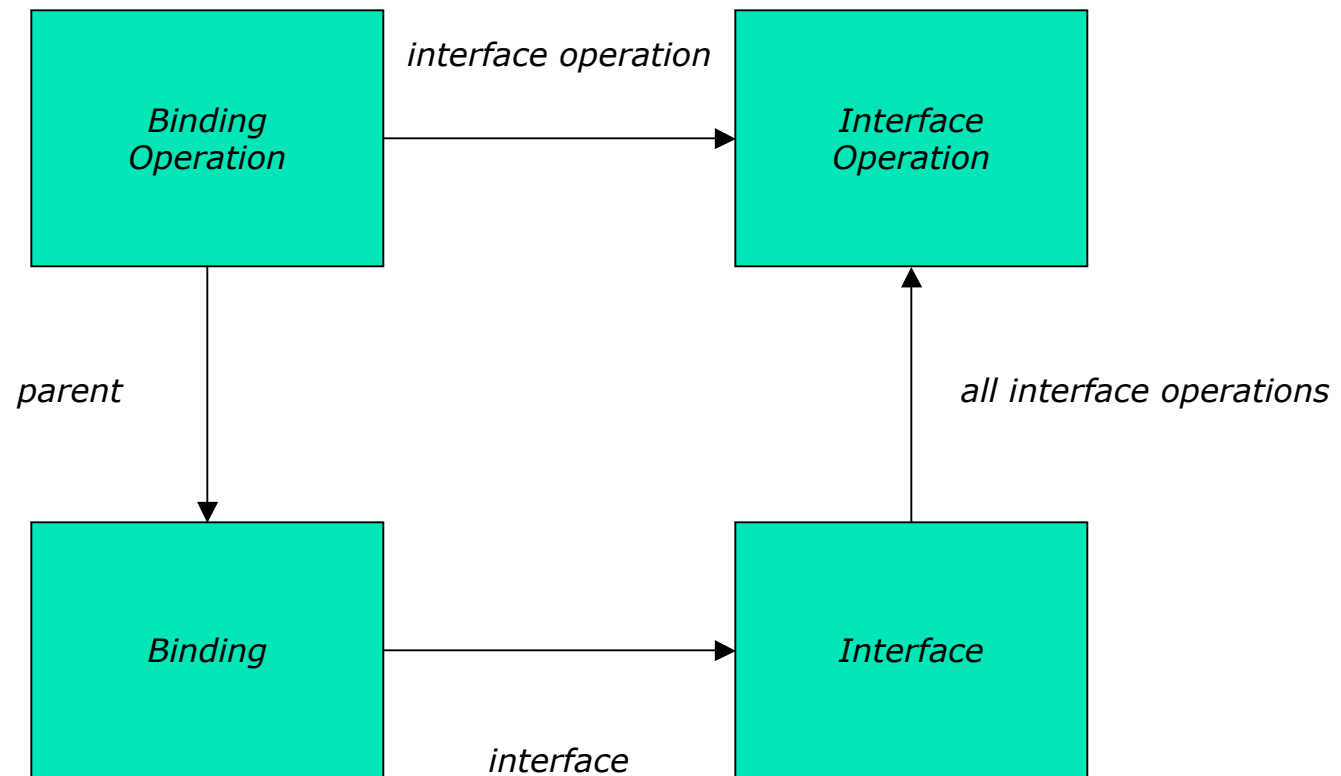


# Binding Operation

---

- Describes binding details for an operation of a service
- Maps to <binding><operation>
- **{interface operation}** REQUIRED. An Interface Operation component in the **{interface operations}** property of the **Interface** component identified by the **{interface}** property of the [parent] **Binding** component, or an Interface component that that **Interface** component directly or indirectly extends. This is the **Interface Operation** component for which binding information is being specified.
- **{binding message references}** OPTIONAL. A set of **Binding Message Reference** components.
- **{binding fault references}** OPTIONAL. A set of **Binding Fault Reference** components.

# Binding Operation Consistency



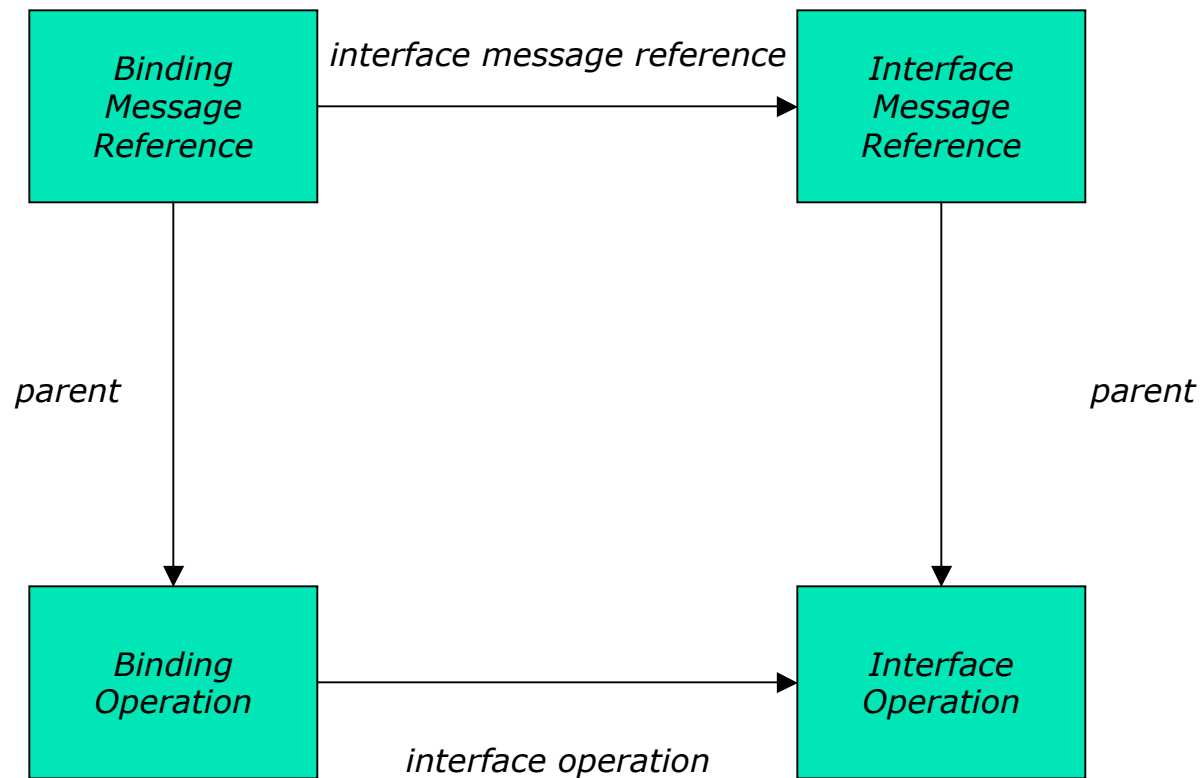


# Binding Message Reference

---

- Describes binding details for a message used by an operation of a service
- Maps to  
<binding><operation>(<input> | <output>)
- **{interface message reference}** REQUIRED. An Interface Message Reference component among those in the {interface message references} property of the Interface Operation component being bound by the containing Binding Operation component.

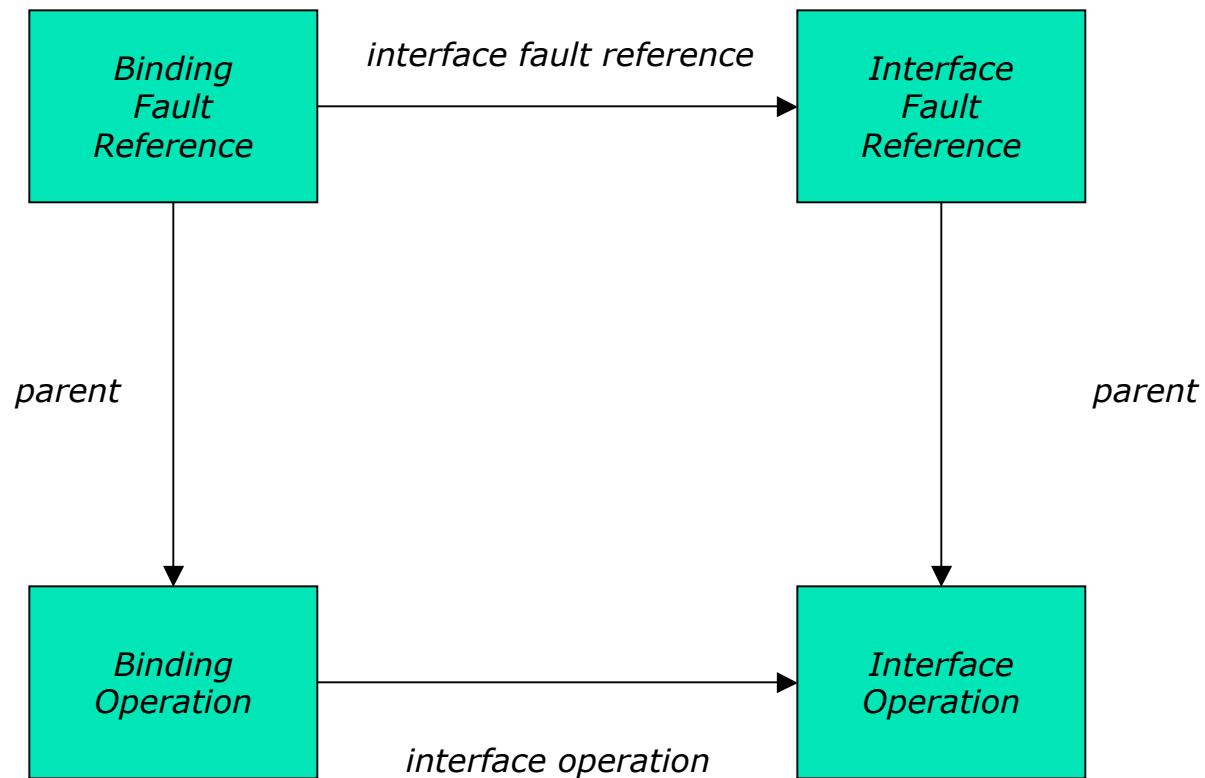
# Binding Message Reference Consistency



# Binding Fault Reference

- Describes binding details for a fault that is thrown by an operation of a service
- Maps to  
<binding><operation>(<infault> | <outfault>)
- **{interface fault reference}** REQUIRED. An Interface Fault Reference component among those in the {interface fault references} property of the Interface Operation component being bound by the parent Binding Operation component.

# Binding Fault Reference Consistency





# Service

---

- Describes the endpoints for accessing the service.
- Maps to `<wsdl:service>`
- **{name}** REQUIRED. An *xs:QName*.
- **{interface}** REQUIRED. An Interface component.
- **{endpoints}** REQUIRED. A non-empty set of Endpoint components.

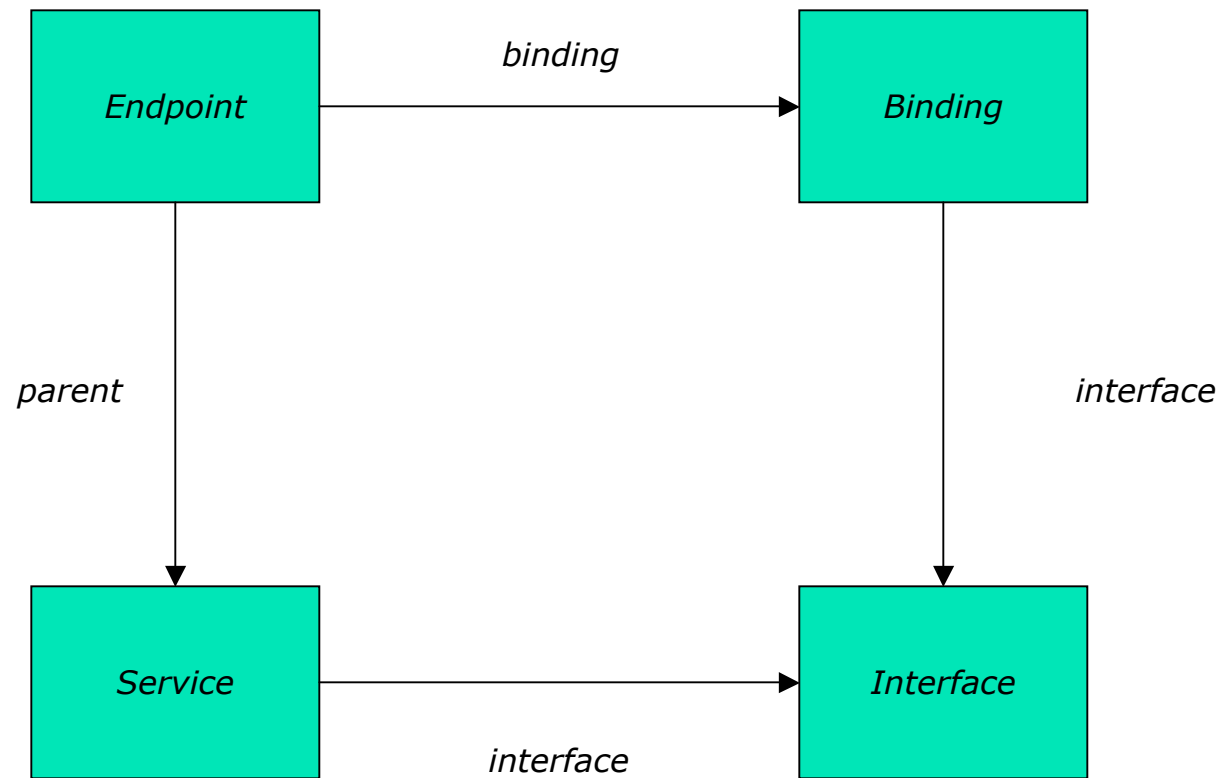


# Endpoint

---

- Describes an endpoint that can be used to access a service using some binding
- Maps to `<service><endpoint>`
- **{name}** REQUIRED. An *xs:NCName*.
- **{binding}** REQUIRED. A Binding component.
- **{address}** OPTIONAL. An *xs:anyURI*. This *xs:anyURI* MUST be an absolute IRI as defined by [IETF RFC 3987]. If present, the value of this attribute represents the network address at which the service indicated by the parent Service component's **{interface}** property is offered *via* the binding referred to by the **{binding}** property.

# Endpoint Consistency





# Call for Participation

---





# Schedule

---

- Second Last Call
  - June 2005 (August 3 actual, ending September 19)
- Candidate Recommendation
  - October 2005
- Proposed Recommendation
  - Early 2006



# Next Steps

---

- Entering W3C Candidate Recommendation phase
  - Please review the spec!
  - Please contribute test cases!
- Two interoperable implementations are required for each feature
  - Please contribute to the Apache and Eclipse implementations!

# Open Source WSDL 2.0 Projects

---

- Apache Woden
- Eclipse Web Tools Platform





# Apache Woden

---

- See: <http://incubator.apache.org/woden/>
- Follow-on to IBM WSDL4J and JSR 110: JWSDL
- Currently an incubator project
- Deliverables:
  - WSDL 2.0 API,
  - model,
  - parser, and
  - validator
- Will be used by Axis2 for code generators:
  - WSDL2Java
  - Java2WSDL



# Eclipse Web Tools Project

---

- See: <http://eclipse.org/webtools/>
- Web Standard Tools subproject
- Integrate Woden into IDE
- Upgrade all tools to support WSDL 2.0:
  - WSDL Validator
  - WSDL Editor
  - WSDL Explorer

# Review the WSDL 2.0 Spec!

Web Services Description Working Group - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://www.w3.org/2002/ws/desc/

Go

Customize Links Free Hotmail Windows Media Windows

**W3C** Architecture domain Web Services Activity

[About Web services](#) · [Web Services Activity statement](#)  
[Administrative page](#) · [Web Services CG](#)

## Web Services Description Working Group

[Charter](#) · [Drafts](#) · [Current Action Items](#) · [Current Editorial Action Items](#) · [Meeting records](#) · [Discussion lists](#) · [Agenda](#) · [Tools](#)  
· [Participants](#)

### Drafts

#### Published W3C Working Drafts

- 2005-08-03. [Web Services Description Language \(WSDL\) Version 2.0 Part 0: Primer](#) (Last Call ends 19 Sep 2005)
- 2005-08-03. [Web Services Description Language \(WSDL\) Version 2.0 Part 1: Core Language and schema](#) (Last Call ends 19 Sep 2005)
- 2005-08-03. [Web Services Description Language \(WSDL\) Version 2.0 Part 2: Adjuncts and SOAP 1.2 binding schema, HTTP binding schema](#) (Last Call ends 19 Sep 2005)
- 2005-08-03. [Web Services Description Language \(WSDL\) Version 2.0 SOAP 1.1 Binding](#) (Last Call ends 19 Sep 2005)

Done



# Contribute to the WSDL 2.0 Test Suite!

2002/ws/desc/test-suite/ - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://dev.w3.org/cvswEB/2002/ws/desc/test-suite/

Go

Customize Links Free Hotmail Windows Media Windows

**W3C** [CVS](#)

## 2002/ws/desc/test-suite/

Click on a directory to enter that directory. Click on a file to display its revision history and to get a chance to display diffs between revisions.

Current directory: [\[Public\]](#) / [2002](#) / [ws](#) / [desc](#) / [test-suite](#)

<a href="#">File</a>	<a href="#">Rev.</a>	<a href="#">Age</a>	<a href="#">Author</a>	<a href="#">Last log entry</a>
<a href="#">Previous Directory</a>				
<a href="#">Attic/</a> [Don't hide]				
<a href="#">documents/</a>				
<a href="#">exchanges/</a>				
<a href="#">messages/</a>				
<a href="#">xmlcatalog/</a>				
<a href="#">build.xml</a>	<a href="#">1.5</a>	2 days	aryman	Added MultipleInlineSchemas-1G for Primer.
<a href="#">index.html</a>	<a href="#">1.4</a>	7 weeks	aryman	Fixed link to WTP project.
<a href="#">catalog.xml</a>	<a href="#">1.2</a>	8 weeks	aryman	Added test cases and Ant script to performan XML schema validation.
<a href="#">TestSuiteContents.html</a>	<a href="#">1.2</a>	4 months	aryman	Added link to W3C Policy for Test Cases.
<a href="#">cvsignore</a>	<a href="#">1.1</a>	10 months	pdowney	initial version

Show only files with tag:  Go

Done



# Contribute to the Apache Woden Project!



# Contribute to the Eclipse WTP Project!

The screenshot shows a Mozilla Firefox browser window displaying the Eclipse Web Tools Platform Project website. The browser's address bar shows the URL <http://eclipse.org/webtools/index.html>. The website features a blue header with the Eclipse logo and the text "eclipse project universal tool platform". A left-hand navigation menu lists various sections such as "eclipse webtools", "WST", "JST", "downloads", "community", "FAQs", "newsgroup", "eclipse home", "about us", "projects", "downloads", "articles", "newsgroups", "community", "search", and "bugs". The main content area is titled "eclipse wtp project" and includes a sub-header "All About The WTP Project". The text describes the project's goal to extend the Eclipse platform with tools for developing J2EE Web applications, listing tools like source editors for HTML, Javascript, CSS, JSP, SQL, XML, DTD, XSD, and WSDL; graphical editors for XSD and WSDL; J2EE project natures, builders, and models; a J2EE navigator; a Web service wizard and explorer; and WS-I Test Tools and database access and query tools and models. Below this text are links for "Getting Started" (instructions on installing Eclipse Web Tools) and "Community" (upcoming events, articles, books, tutorials, and presentations). On the right side, there are sections for "Help?" (with links to FAQ, Newsgroup, and Tutorials) and "Download" (with links to WTP 0.7, What's New, Milestone Plan, and All Downloads). The Milestone Plan section notes that the WTP milestone plan is available [here](#). The browser's status bar at the bottom shows "Done".

