WSDL 1.1 Element Identifiers

W3C Working Draft 31 January 2007

This version:
http://www.w3.org/TR/2007/WD-wsdl11elementidentifiers-20070131
Latest version:
http://www.w3.org/TR/wsdl11elementidentifiers
Editors:
David Orchard, BEA Systems
Asir S Vedamuthu, Microsoft Corporation
Frederick Hirsch, Nokia
Maryann Hondo, IBM Corporation
Prasad Yendluri, webMethods, Inc.
Toufic Boubez, Layer 7 Technologies
Ümit Yalçinalp, SAP AG.

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

Abstract

WSDL 1.1 Element Identifiers defines a syntax to identify individual elements in a WSDL 1.1 document.

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 First Public Working Draft of the WSDL 1.1 Element Identifiers specification. This Working Draft was produced by the members of the Web Services Policy Working Group, which is part of the W3C Web Services Activity. The Working Group expects to advance this Working Draft to Working Group Note.

Several issues have been filed on this document and are recorded in Bugzilla. The Working Group solicits feedback on a proposal to change the syntax of some WSDL 1.1. element identifiers.

Note that this Working Draft does not necessarily represent a consensus of the Working Group. Discussion of this document takes place on the public public-ws-policy@w3.org mailing list (public archive) and within Bugzilla. Comments on this specification should be made following the Working Group's process for submitting issues.

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. The group does not expect this document to become a W3C Recommendation. 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

Appendix

---

1. Introduction

This document defines an element identifier syntax for WSDL 1.1 elements.

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 [*RFC 2119 [p.9]*].

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

2. Fragment Identifiers

This section defines a fragment identifier syntax for identifying elements of a WSDL 1.1 document. This fragment identifier syntax is compliant with the [*XPointer Framework [p.9]*]. This document is primarily based upon [*WSDL 2.0 Core [p.8]*]. There is a substantial difference between the WSDL 1.1 and WSDL 2.0 fragment identifiers.WSDL 2.0 defines fragment identifiers with respect to the WSDL 2.0 component model, whereas WSDL 1.1 defines XML element and attribute syntax only. Because there is no WSDL 1.1 component model, the WSDL 1.1 fragment identifiers are to the WSDL 1.1 elements. Further, the fragment identifers are to the WSDL 1.1 elements prior to any processing of the WSDL document, such as validation, inclusion, imports, schema type validation, etc.

A WSDL 1.1 fragment identifier is an XPointer [*XPointer Framework [p.9]*], augmented with WSDL 1.1 pointer parts as defined below. Note that many of these parts require the pre-appearance of one or more `xmlns` pointer parts (see 3.4 Namespace Binding Context in [*XPointer Framework [p.9]*]). The pointer parts have a scheme name that corresponds to one of the standard WSDL 1.1 element names, and scheme data that is a path composed of names that identify the elements. The scheme names all begin with the prefix "wsdl11." to avoid name conflicts with other schemes. The names in the path are of type either QName, NCName, IRI, URI, or Pointer Part depending on the context. The scheme data for WSDL 1.1 extension elements is defined by the corresponding extension specification.

For QNames, any prefix MUST be defined by a preceding xmlns pointer part. If a QName does not have a prefix then its namespace name is the target namespace of the WSDL 1.1 document.

The fragment identifier is typically constructed from the `name` property of the element and the `name` properties of its ancestors as a path according to Table 2-1 [p.3] . The first column of this table gives the name of the WSDL 1.1 element. Columns labeled 1 through 3 specify the identifiers that uniquely identify the element within its context. Identifiers are typically formed from the `name` property, although in several cases references to other elements are used. These identifiers are then used to construct the pointer part in the last column. The fragment identifier in a WSDL 1.1 element IRI-reference MUST resolve to some element as defined by the construction rules in Table 2-1 [p.3] .

Table 2-1. Rules for determining pointer parts for WSDL 1.1 elements

element	1	2	3	Pointer Part
Definitions	n/a	n/a	n/a	`wsdl11.definitions()`
Type Definition	`types` QName	n/a	n/a	`wsdl11.types(types)`
Element Declaration	`element` QName	n/a	n/a	`wsdl11.elementDeclaration(element)`
Message	`message` NCName	n/a	n/a	`wsdl11.message(message)`
Message Part	`message` NCName	`part` NCName	n/a	`wsdl11.messagePart(message/part)`
portType	`portType` NCName	n/a	n/a	`wsdl11.portType(portType)`
portType Operation	`portType` NCName	`operation` NCName	n/a	`wsdl11.portTypeOperation(portType/operation)`
portType Message Reference	`portType` NCName	`operation` NCName	`message` NCName	`wsdl11.portTypeMessageReference(portType/operation/message)`
portType Operation Fault	`portType` NCName	`operation` NCName	`fault` QName	`wsdl11.portTypeOperationFault(portType/operation/fault)`
Binding	`binding` NCName	n/a	n/a	`wsdl11.binding(binding)`
Binding Operation	`binding` NCName	`operation` QName	n/a	`wsdl11.bindingOperation(binding/operation)`
Binding Message Reference	`binding` NCName	`operation` QName	`message` NCName	`wsdl11.bindingMessageReference(binding/operation/message)`
Binding Operation Fault	`binding` NCName	`operation` QName	`fault` NCName	`wsdl11.bindingOperationFault(binding/operation/fault)`
Service	`service` NCName	n/a	n/a	`wsdl11.service(service)`
port	`service` NCName	`port` NCName	n/a	`wsdl11.port(service/port)`

Extensions	`namespace` URI	`identifier` extension-specific-syntax	n/a	`wsdl11.extension(namespace,identifier)`

3. IRI-References for WSDL 1.1 Elements

This section provides a syntax for IRI-references for all elements found in a [*WSDL 1.1 [p.9]*] document. The IRI-references are easy to understand and compare, while imposing no burden on the WSDL 1.1 author.

3.1 WSDL 1.1 IRIs

There are two main cases for WSDL 1.1 IRIs:

- the IRI of a WSDL 1.1 document
- the IRI of a WSDL 1.1 namespace

The IRI of a WSDL 1.1 document can be dereferenced to give a resource representation that contributes elements to a single WSDL 1.1 namespace. If the media type is set to the WSDL 1.1 media type i.e. application/xml, then the fragment identifiers can be used to identify the main elements that are defined in the document.

In keeping with WSDL 1.1, which has a recommendation that that the namespace URI be dereferencible to a WSDL 1.1 document, this section specifies the use of the namespace IRI with the WSDL 1.1 fragment identifiers to form an IRI-reference.

The IRI in an IRI-reference for a WSDL 1.1 element is the namespace name of the `name` property of either the element itself, in the case of `portType` , `Binding` , and `Service` elements, or the `name` property of the ancestor top-level element. The IRI provided by the namespace name of the `name` property is combined with a zero or more `xmlns` pointer parts (see 3.4 Namespace Binding Context in [*XPointer Framework [p.9]*]) followed by a single WSDL 1.1 pointer part, following the same rules as defined for WSDL 1.1 fragment ids **2. Fragment Identifiers** [p.3] .

3.2 IRI Identification Of SOAP Binding elements

`SOAP Binding` elements (binding, operation, body, header, fault, headerfault, and address) can be identified using the *wsdl11.extension* XPointer Framework scheme according to the following rules:

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.binding(parent))`, where:

- *`parent`* is the pointer part of the `SOAP Binding's parent` element

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.operation(parent))`

- *`parent`* is the pointer part of the `SOAP Operation's parent` element

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.body(parent))`

- *`parent`* is the pointer part of the `SOAP Body's parent` element

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.header(parent))`

- *`parent`* is the pointer part of the `SOAP Header's parent` element

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.headerfault(parent))`

- *`parent`* is the pointer part of the `SOAP HeaderFault's parent` element

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.fault(parent))`

- *`parent`* is the pointer part of the `SOAP Fault's parent` element

`wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/, w11soap.address(parent))`

- *`parent`* is the pointer part of the `SOAP Address's parent` element

3.3 Canonical Form for WSDL 1.1 element identifiers

The IRI-references described above MAY be used as WSDL 1.1 element identifiers. For ease of comparison, the fragment identifier of WSDL 1.1 element identifiers SHOULD conform to the following canonicalization rules:

- The fragment identifier consists of a sequence zero or more `xmlns()` pointer parts followed by exactly one `wsdl1.*()` pointer part.
- Each `xmlns()` pointer part that appears in the fragment identifier defines a namespace that is referenced by the `wsdl1.*()` pointer part.
- Each `xmlns()` pointer part defines a unique namespace.
- The `xmlns()` pointer parts define namespaces in the same order as they are referenced in the `wsdl1.*()` pointer part.
- The namespace prefixes defined by the `xmlns()` pointer parts are named `ns1` , `ns2` , etc., in the order of their appearance.
- The fragment identifier contains no optional whitespace.

3.4 Example

Consider WSDL 1.1 document located at http://example.org/TicketAgent.wsdl. Each WSDL 1.1 Element Identifier is shown in comments above the WSDL 1.1 element

Example 3-1. IRI-References - Example WSDL 1.1 Document

```
<?xml version="1.0" encoding="UTF-8"?>

<wsdl:definitions targetNamespace="http://example.org/TicketAgent.wsdl11"
    xmlns:tns="http://example.org/TicketAgent.wsdl11"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsTicketAgent="http://example.org/TicketAgent.xsd"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xsi:schemaLocation="http://schemas.xmlsoap.org/wsdl/ http://www.w3.org/TR/2001/NOTE-wsdl-20010315/wsdl11.xsd">

    <!-- http://example.org/TicketAgent.wsdl11#wsdl11.definitions() -->

    <wsdl:types>
        <xs:schema xmlns:xsTicketAgent="http://example.org/TicketAgent.xsd"
            targetNamespace="http://example.org/TicketAgent.xsd">
            <xs:element name="listFlightsRequest" type="xsTicketAgent:tListFlights"/>
            <!-- Starting from here, http://example.org/TicketAgent.wsdl11 will be shortened to http://...
         http://...#xmlns(ns1=http://example.org/TicketAgent.xsd)
        wsdl11.elementDeclaration(ns1:listFlightsRequest) -->
            <xs:complexType name="tListFlights">
                <xs:sequence>
                    <xs:element name="travelDate" type="xs:date"/>
                    <xs:element name="startCity" type="xs:string"/>
                    <xs:element name="endCity" type="xs:string"/>
                </xs:sequence>
            </xs:complexType>
            <xs:element name="listFlightsResponse" type="xsTicketAgent:tFlightsResponse"/>
            <!-- http://...#xmlns(ns1=http://example.org/TicketAgent.xsd)
         wsdl11.elementDeclaration(ns1:listFlightsResponse) -->
            <xs:complexType name="tFlightsResponse">
                <xs:sequence>
                    <xs:element name="flightNumber" type="xs:integer" minOccurs="0"
                        maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:schema>
    </wsdl:types>

    <wsdl:message name="listFlightsRequest">
```

```
        <!-- http://...#wsdl11.message(listFlightsRequest) -->
        <wsdl:part name="body" element="xsTicketAgent:listFlightsRequest"/>
        <!-- http://...#wsdl11.messagePart(listFlightsRequest/body) -->
    </wsdl:message>

    <wsdl:message name="listFlightsResponse">
        <!-- http://...#wsdl11.message(listFlightsResponse) -->
        <wsdl:part name="body" element="xsTicketAgent:listFlightsResponse"/>
        <!-- http://...#wsdl11.messagePart(listFlightsResponse/body) -->
    </wsdl:message>

    <wsdl:portType name="TicketAgent">
        <!-- http://...#wsdl11.portType(TicketAgent) -->
        <wsdl:operation name="listFlights">
            <!-- http://...#wsdl11.portTypeOperation(TicketAgent/listFlights) -->
            <wsdl:input message="tns:listFlightsRequest"/>
            <!-- http://...#wsdl11.portTypeMessageReference(TicketAgent/listFlights/input) -->
            <wsdl:output message="tns:listFlightsResponse"/>
            <!-- http://...#wsdl11.portTypeMessageReference(TicketAgent/listFlights/output) -->
        </wsdl:operation>
    </wsdl:portType>

    <wsdl:binding name="TicketAgentSoap" type="tns:TicketAgent">
        <!-- http://...#wsdl11.binding(TicketAgentSoap) -->
        <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
        <!-- http://...#wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/,
             w11soap.binding( wsdl11.binding(TicketAgentSoap)) -->
        <wsdl:operation name="listFlights">
            <!-- http://...#wsdl11.bindingOperation(TicketAgentSoap/listFlights) -->
            <wsdl:input>
                <!-- http://...#wsdl11.bindingOperationMessageReference(TicketAgentSoap/listFlights/input) -->
                <soap:body parts="body" use="literal"/>
                <!-- http://...#wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/,
                          w11soap.body(wsdl11.bindingOperationMessageReference
                                      (TicketAgentSoap/listFlights/input)) -->
            </wsdl:input>
            <wsdl:output>
                <!-- http://...#wsdl11.bindingOperationMessageReference(TicketAgentSoap/listFlights/output) -->
                <soap:body parts="body" use="literal"/>
                <!-- http://...#wsdl11.extension(http://schemas.xmlsoap.org/wsdl/soap/,
                          w11soap.body(wsdl11.bindingOperationMessageReference
                                      (TicketAgentSoap/listFlights/output)) -->
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>

</wsdl:definitions>
```

4. References

4.1 Normative References

[RFC 3023]
IETF "RFC 3023: XML Media Types", M. Murata, S. St. Laurent, D. Kohn, July 1998. (See *http://www.ietf.org/rfc/rfc3023.txt*.)

[WSDL 2.0 Core]
Web Services definitions Language (WSDL) Version 2.0 Part 1: Core Language , R. Chinnici, J-J. Moreau, A. Ryman, S. Weerawarana, Editors. W3C Candidate Recommendation 27 March 2006. The current version of Web Services definitions Language (WSDL) Version 2.0 Part 1: Core Language is available at http://www.w3.org/TR/2006/CR-wsdl20-20060327 . The latest version of "Web Services definitions Language (WSDL) Version 2.0 Part 1: Core Language" is available at http://www.w3.org/TR/wsdl20.

[WSDL 1.1]
Web Services definitions Language (WSDL) 1.1, E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, Authors. W3C Note 15 March 2002. The current version of Web Services Description Language (WSDL) 1.1 is available at http://www.w3.org/TR/2001/NOTE-wsdl-20010315 . The latest version of Web Services definitions Language 1.1 is available at http://www.w3.org/TR/wsdl11.

[XPointer Framework]
XPointer Framework,Paul Grosso, Eve Maler, Jonathan Marsh, Norman Walsh, Editors. W3C Recommendation 25 March 2003. The current version of XPointer Framework is available at http://www.w3.org/TR/2003/REC-xptr-framework-20030325/ . The latest version of XPointer Framework is available at http://www.w3.org/TR/xptr-framework/.

[RFC 2119]
IETF "RFC 2119: Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. (See *http://www.ietf.org/rfc/rfc2119.txt*.)

[RFC 3986]
IETF "RFC 3986: Uniform Resource Identifiers (URI): Generic Syntax", T. Berners-Lee, R. Fielding, L. Masinter, January 2005. (See *http://www.ietf.org/rfc/rfc3986.txt*.)

A. Change Log (Non-Normative)

Table A-1. Changes

Who	When	What
DBO	20061108	Initial Revision
DBO	20061212	Uncommented canonical section, fixed editorial items
DBO	20070122	Resolution of bug 4208, AI is 145
FS	20070127	Editorial fixes for publication preparation