



# Service Component Architecture Web Service Binding Specification Version 1.1

Committee Draft 03 Revision 4 Proposal for Issue 126, v1

22 April 2010

**Specification URIs:**

**This Version:**

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd03.html>  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd03.doc>  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd03.pdf> (Authoritative)

**Previous Version:**

<http://www.oasis-open.org/committees/download.php/31235/sca-binding-ws-1.1-spec-cd02.doc>  
<http://www.oasis-open.org/committees/download.php/31236/sca-binding-ws-1.1-spec-cd02.pdf> (Authoritative)

**Latest Version:**

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec.html>  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec.doc>  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec.pdf> (Authoritative)

**Technical Committee:**

OASIS Service Component Architecture / Bindings (SCA-Bindings) TC

**Chair(s):**

Simon Holdsworth, IBM

**Editor(s):**

Simon Holdsworth, IBM  
Anish Karmarkar, Oracle  
Piotr Przybylski, IBM

**Related work:**

This specification replaces or supersedes:

- Service Component Architecture Web Service Binding Specification Version 1.00, March 21 2007

This specification is related to:

- OASIS Committee Draft 03, "Service Component Architecture Assembly Model Specification Version 1.1", March 2009  
<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-cd03.pdf>
- OASIS Committee Draft 02, "SCA Policy Framework Version 1.1", February 2009  
<http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd02.pdf>

**Declared XML Namespace(s):**

<http://docs.oasis-open.org/ns/opencsa/sca/200912>

**Abstract:**

The SCA Web Service binding specified in this document applies to the services and references of an SCA composite [**SCA-Assembly**]. It defines the manner in which a service can be made available as a web service, and in which a reference can invoke a web service.

This binding is a WSDL-based binding; that means it either references an existing WSDL binding or specifies enough information to generate one. When an existing WSDL binding is not referenced, rules defined in this document specify how to generate a WSDL binding.

**Status:**

This document was last revised or approved by the OASIS Service Component Architecture / Bindings (SCA-Bindings) TC on the above date. The level of approval is also listed above. Check the "Latest Version" or "Latest Approved Version" location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at <http://www.oasis-open.org/committees/sca-bindings/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<http://www.oasis-open.org/committees/sca-bindings/ipr.php>).

The non-normative errata page for this specification is located at <http://www.oasis-open.org/committees/sca-bindings/>.

---

## Notices

Copyright © OASIS® 2005, 2010. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" "SCA" and "Service Component Architecture" are trademarks of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

## Table of Contents

1	Introduction	6
1.1	Terminology	6
1.2	Normative References	7
1.3	Non-Normative References	8
1.4	Naming Conventions	8
2	Web Service Binding Schema	9
2.1	Compatibility of SCA Service Interfaces and WSDL portTypes	11
2.2	Endpoint URI resolution	11
2.3	Interface mapping	<del>11</del> 1142
2.4	Production of WSDL description for an SCA service	12
2.5	Additional binding configuration data	12
2.6	Web Service Binding and SOAP Intermediaries	<del>12</del> 1243
2.7	Support for WSDL extensibility	<del>12</del> 1243
2.8	Intents listed in the bindingType	<del>12</del> 1243
2.9	Intents and binding configuration	13
3	Web Service Binding Examples	<del>13</del> 1445
3.1	Example Using WSDL documents	<del>13</del> 1445
3.2	Examples Without a WSDL Document	<del>13</del> 1445
4	Transport Binding	<del>14</del> 1647
4.1	Intents	<del>14</del> 1647
4.2	Default Transport Binding Rules	<del>14</del> 1647
4.2.1	WS-I Basic Profile Alignment	<del>14</del> 1647
4.2.2	Default Transport Binding Rules	<del>14</del> 1647
5	Implementing SCA Callbacks using Web Services	<del>15</del> 1819
5.1	SCA Web Services Callback Protocol	<del>15</del> 1819
5.2	SCA Web Services Callback Protocol with WS-MakeConnection	<del>15</del> 1920
5.3	Policy Assertion for SCA Web Services Callback Protocol	<del>15</del> 1920
5.3.1	Assertion Model	<del>15</del> 1924
5.3.2	Normative Outline	<del>15</del> 1924
5.3.3	Assertion Attachment	<del>15</del> 2024
5.3.4	Assertion Example	<del>15</del> 2024
5.3.5	Security Considerations	<del>15</del> 2122
6	Conformance	<del>16</del> 2223
6.1	SCA WS Binding XML Document	<del>16</del> 2223
6.2	SCA Runtime	<del>16</del> 2223
A.	Web Services XML Binding Schema: sca-binding-webservice.xsd	<del>16</del> 2425
B.	SCA Web Services Callback Protocol Policy Assertion XML Schema: sca-binding-webservice-callback.xsd	<del>16</del> 2526
C.	Conformance Items	<del>16</del> 2627
D.	WSDL Generation	<del>16</del> 3031
E.	SCA Web Services Callback Protocol Message Examples	<del>16</del> 3132
E.1	Message Examples Using WS-MakeConnection	<del>16</del> 3334
F.	Acknowledgements	<del>16</del> 3536

| G. Revision History.....~~36~~37

# 1 Introduction

The SCA Web Service binding specified in this document applies to the services and references of composites and components [**SCA-Assembly**]. It defines the manner in which a service can be made available as a web service, and in which a reference can invoke a web service.

This binding is a WSDL-based binding; that means it either references an existing WSDL binding or can be configured to specify enough information to generate one. When an existing WSDL binding is not referenced, rules defined in this document specify how to generate a WSDL binding. This specification only defines a binding using WSDL 1.1.

The Web Service binding can point to an existing WSDL [**WSDL11**] document, separately authored, that specifies the details of the WSDL binding to be used to provide or invoke the web service. In this case the SCA web services binding allows anything that is valid in a WSDL binding, including rpc-encoded style and binding extensions. It is the responsibility of the SCA system provider to ensure support for all options specified in the WSDL binding. Interoperation of such services is not guaranteed.

The SCA Web Service binding also provides attributes that can be used to provide the details of a WSDL SOAP binding. This allows a WSDL document to be synthesized in the case that one does not already exist. In this case only WS-I compliant mapping is supported.

The SCA Web Service binding can be further customized through the use of SCA Policy Sets. For example, a requirement to conform to a WS-I profile [**WSI-Profiles**] could be represented with a policy set.

## 1.1 Terminology

The key words "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**].

This specification uses predefined namespace prefixes throughout; they are given in the following list. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

Prefix	Namespace	Notes
xs	"http://www.w3.org/2001/XMLSchema"	Defined by XML Schema 1.0 specification
wsa	"http://www.w3.org/2005/08/addressing"	Defined by WS-Addressing 1.0
wsp	"http://www.w3.org/ns/ws-policy"	Defined by WS-Policy 1.5
wsrmp	"http://docs.oasis-open.org/ws-rx/wsrmp/200702"	Defined by WS-ReliableMessaging Policy 1.2
soap11	"http://schemas.xmlsoap.org/soap/envelope/"	Defined by SOAP 1.1
soap12	"http://www.w3.org/2005/08/addressing"	Defined by SOAP 1.2
wsdli	"http://www.w3.org/ns/wsdli-instance"	Defined by WSDL 2.0
wsoap11	"http://schemas.xmlsoap.org/wsdli/soap/"	Defined by WSDL 1.1 [WSDL11]
wsoap12	"http://schemas.xmlsoap.org/wsdli/soap12/"	Defined by [W11-SOAP12]
sca	"http://docs.oasis-open.org/ns/opencsa/sca/200912"	Defined by the SCA specifications

28 **1.2 Normative References**

29	<b>[RFC2119]</b>	S. Bradner, <i>Key words for use in RFCs to Indicate Requirement Levels</i> , <a href="http://www.ietf.org/rfc/rfc2119.txt">http://www.ietf.org/rfc/rfc2119.txt</a> , IETF RFC 2119, March 1997.
30		
31	<b>[SCA-Assembly]</b>	OASIS Committee Draft 03, "Service Component Architecture Assembly Model Specification Version 1.1", March 2009 <a href="http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-cd03.pdf">http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-cd03.pdf</a>
32		
33		
34		
35	<b>[SCA-Policy]</b>	OASIS Committee Draft 02, "SCA Policy Framework Specification Version 1.1", February 2009 <a href="http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd02.pdf">http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd02.pdf</a>
36		
37		
38	<b>[SCA-JCAA]</b>	OASIS Committee Draft 03, "SCA Java Common Annotations and APIs Specification Version 1.1", May 2009 <a href="http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec-cd03.pdf">http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec-cd03.pdf</a>
39		
40		
41	<b>[WSDL11]</b>	E. Christensen et al, <i>Web Service Description Language (WSDL) 1.1</i> , <a href="http://www.w3.org/TR/2001/NOTE-wsdl-20010315">http://www.w3.org/TR/2001/NOTE-wsdl-20010315</a> , W3C Note, March 15 2001.
42		
43	<b>[WSDL20]</b>	Chinnici et al, <i>Web Service Description Language (WSDL) Version 2.0 Part 1:  Core Language</i> , <a href="http://www.w3.org/TR/2007/REC-wsdl20-20070626/">http://www.w3.org/TR/2007/REC-wsdl20-20070626/</a> , W3C Recommendation, June 26 2007.
44		
45		
46	<b>[WSI-Profiles]</b>	"Basic Profile Version 1.1" <a href="http://www.ws-i.org/Profiles/BasicProfile-1.1.html">http://www.ws-i.org/Profiles/BasicProfile-1.1.html</a> , "Attachments Profile Version 1.0" <a href="http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html">http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html</a> , "Simple SOAP Binding Profile Version 1.0" <a href="http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html">http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html</a> , "Basic Security Profile Version 1.0" <a href="http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html">http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html</a>
47		
48		
49		
50		
51		
52		
53	<b>[JAX-WS]</b>	"JSR 224: Java™ API for XML-Based Web Services (JAX-WS) 2.0" <a href="http://jcp.org/en/jsr/detail?id=224">http://jcp.org/en/jsr/detail?id=224</a>
54		
55	<b>[SOAP11]</b>	Box et al, "Simple Object Access Protocol (SOAP) 1.1" <a href="http://www.w3.org/TR/2000/NOTE-SOAP-20000508/">http://www.w3.org/TR/2000/NOTE-SOAP-20000508/</a> , W3C Note May 2000
56		
57	<b>[SOAP]</b>	Gudgin et al, "SOAP Version 1.2 Part 1: Messaging Framework" <a href="http://www.w3.org/TR/2003/REC-soap12-part1-20030624/">http://www.w3.org/TR/2003/REC-soap12-part1-20030624/</a> , W3C Recommendation June 2003; Box et al, "Simple Object Access Protocol (SOAP) 1.1" <a href="http://www.w3.org/TR/2000/NOTE-SOAP-20000508/">http://www.w3.org/TR/2000/NOTE-SOAP-20000508/</a> , W3C Note May 2000
58		
59		
60		
61	<b>[SOAP12Adjuncts]</b>	Gudgin et al, "SOAP Version 1.2 Part 2: Adjuncts (Second Edition)" <a href="http://www.w3.org/TR/soap12-part2/">http://www.w3.org/TR/soap12-part2/</a> , W3C Recommendation April 2007
62		
63	<b>[WS-Addr]</b>	Gudgin et al, "Web Services Addressing 1.0 – Core" <a href="http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/">http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/</a> , W3C Recommendation May 2006
64		
65		
66	<b>[W11-SOAP12]</b>	Angelov et al, "WSDL 1.1 Binding Extension for SOAP 1.2" <a href="http://www.w3.org/Submission/wsdl11soap12/">http://www.w3.org/Submission/wsdl11soap12/</a> , W3C Member Submission April 2006
67		
68		
69	<b>[WS-Addr-SOAP]</b>	Gudgin et al, "Web Services Addressing 1.0 – SOAP Binding" <a href="http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/">http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/</a> , W3C Recommendation May 2006
70		
71		
72	<b>[WS-Addr-Meta]</b>	<b>Gudgin et al, "Web Services Addressing 1.0 – Metadata"</b> <a href="http://www.w3.org/TR/2007/REC-ws-addr-metadata-20070904/">http://www.w3.org/TR/2007/REC-ws-addr-metadata-20070904/</a> , W3C <b>Recommendation September 2007</b>
73		
74		
75	<b>[WS-MC]</b>	OASIS Standard "Web Services Make Connection (WS-MakeConnection) Version 1.1", February 2009 <a href="http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.doc">http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.doc</a>
76		
77		

78       **[WS-Policy]**       Vedamuthu et al, "Web Services Policy 1.5 – Framework"  
79                           <http://www.w3.org/TR/2007/REC-ws-policy-20070904>, W3C Recommendation  
80                           September 2007  
81       **[WS-PA]**        Vedamuthu et al, "Web Services Policy 1.5 – Attachment"  
82                           <http://www.w3.org/TR/2007/REC-ws-policy-attach-20070904>, W3C  
83                           Recommendation September 2007

### 84   **1.3 Non-Normative References**

85       **[WSI-AP]**        "Attachments Profile Version 1.0" [http://www.w3-](http://www.w3.org/Profiles/AttachmentsProfile-1.0.html)  
86                           [i.org/Profiles/AttachmentsProfile-1.0.html](http://www.w3.org/Profiles/AttachmentsProfile-1.0.html)  
87       **[MTOM]**        Gudgin et al, "SOAP Message Transmission Optimization Mechanism"  
88                           <http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/>, W3C  
89                           Recommendation January 2005  
90       **[WS-Security]**   Oasis Standard "Web Services Security: SOAP Message Security 1.1 (WS-  
91                           Security 2004)" February 2006 [http://docs.oasis-open.org/wss/v1.1/wss-v1.1-](http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)  
92                           [spec-os-SOAPMessageSecurity.pdf](http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)

### 93   **1.4 Naming Conventions**

94   The naming conventions used by artefacts defined in this specification are:

- 95   • The naming conventions defined by section 1.3 of the Assembly Specification **[SCA-Assembly]**.
- 96   • Where the names of elements and attributes consist partially or wholly of acronyms, the letters of the  
97    acronyms use the same case. When the acronym appears at the start of the name of an element or  
98    an attribute, or after a period, it is in lower case. If it appears elsewhere in the name of an element or  
99    an attribute, it is in upper case. For example, an attribute might be named "uri" or "jndiURL".
- 100   • Where the names of types consist partially or wholly of acronyms, the letters of the acronyms are in  
101    all upper case. For example, an XML Schema type might be named "JCABinding" or "MessageID".
- 102   • Values, including local parts of QName values, follow the rules for names of elements and attributes  
103    as stated above, with the exception that the letters of acronyms are in all upper case. For example, a  
104    value might be "JMSDefault" or "namespaceURI".



## 2 Web Service Binding Schema

The Web Service binding element is defined by the pseudo-schema in Snippet 2-1.

```
<binding.ws name="xs:NCName"?
  requires="list of xs:QName"?
  policySets="list of xs:QName"?
  uri="xs:anyURI"?
  wsdlElement="xs:anyURI"?
  wsdl:wsdlLocation="list of xs:anyURI pairs"? >
  <wireFormat ... />?
  <operationSelector ... />?
  <endpointReference>...</endpointReference>*
</binding.ws>
```

Snippet 2-1: binding.ws Pseudo-Schema

The **binding.ws** element has the attributes:

- **/binding.ws/@name** - as defined in the SCA Assembly Specification [SCA-Assembly].
- **/binding.ws/@requires** - as defined in the SCA Assembly Specification [SCA-Assembly].
- **/binding.ws/@policySets** - as defined in the SCA Assembly Specification [SCA-Assembly].
- **/binding.ws/@uri** - the resolution algorithm of Section 2.2 describes how this attribute is interpreted. For an SCA reference, the @uri attribute MUST be an absolute value. [BWS20001]
- **/binding.ws/@wsdlElement** – when present this attribute specifies the URI of a WSDL element. The value of the @wsdlElement attribute MUST identify an element in an existing WSDL 1.1 document. [BWS20002] The URI can have the following forms:
  - Service:

```
<WSDL-namespace-URI>#wsdl.service(<service-name>)
```

If the binding is for an SCA service, the wsdlElement attribute MUST NOT specify the wsdl.service form of URI. If the binding is for an SCA service, the wsdlElement attribute MUST NOT specify the wsdl.service form of URI. [BWS20003]

If the binding is for an SCA reference, the set of available ports for the reference consists of the ports in the WSDL service that have portTypes which are compatible supersets of the SCA reference as defined in the SCA Assembly Model specification [SCA-Assembly] and satisfy all the policy constraints of the binding.

If the wsdl.service form of wsdlElement is used on an SCA reference binding, the set of available ports for that reference binding MUST be non-empty. If the wsdl.service form of wsdlElement is used on an SCA reference binding, the set of available ports for that reference binding MUST be non-empty. [BWS20004] The set of available ports represents a single SCA reference binding with respect to the multiplicity of that SCA reference. If the wsdl.service form of wsdlElement is used on an SCA reference binding, the SCA runtime MUST raise an error if there are no available ports that it supports. If the wsdl.service form of wsdlElement is used on an SCA reference binding, the SCA runtime MUST raise an error if there are no available ports that it supports. [BWS20005] When an invocation is made using an SCA reference binding with the wsdl.service form of wsdlElement, the SCA runtime MUST use exactly one port from the set of available ports for the reference (with port selection on a per-invocation basis permitted). When an invocation is made using an SCA reference binding with the wsdl.service form of wsdlElement, the SCA runtime MUST use exactly one port from the set of available ports for the reference (with port selection on a per-invocation basis permitted). [BWS20006]

Formatted: Highlight

153 – Port:  
154 <WSDL-namespace-URI>#wsdl.port(<service-name>/<port-name>)

155 If the binding is for an SCA service, the portType associated with the specified WSDL port MUST  
156 be compatible with the SCA service interface as defined in section 2.1, and the port MUST satisfy  
157 all the policy constraints of the binding. [BWS20007] The SCA runtime MUST expose an endpoint  
158 for the specified WSDL port, or raise an error if it does not support the WSDL port. [BWS20008] If  
159 the binding is for an SCA reference, the portType associated with the specified WSDL port MUST  
160 be a compatible superset of the SCA reference interface as defined in the SCA Assembly Model  
161 specification [SCA-Assembly][SCA-Assembly], and the port MUST satisfy all the policy  
162 constraints of the binding. If the binding is for an SCA reference, the portType associated with the  
163 specified WSDL port MUST be a compatible superset of the SCA reference interface as defined  
164 in the SCA Assembly Model specification [SCA-Assembly], and the port MUST satisfy all the  
165 policy constraints of the binding. [BWS20009] The SCA runtime MUST use the specified WSDL  
166 port for invocations made using the SCA reference binding, or raise an error if it does not support  
167 the WSDL port. The SCA runtime MUST use the specified WSDL port for invocations made using  
168 the SCA reference binding, or raise an error if it does not support the WSDL port. [BWS20010]

Formatted: Highlight

169 – Binding:  
170 <WSDL-namespace-URI>#wsdl.binding(<binding-name>)

171 If the binding is for an SCA service, the portType associated with the specified WSDL binding  
172 MUST be compatible with the SCA service interface as defined in section 2.1, and the WSDL  
173 binding MUST satisfy all the policy constraints of the binding. [BWS20011] The SCA runtime  
174 MUST expose an endpoint for the specified WSDL binding, or raise an error if it does not support  
175 the WSDL binding. [BWS20012]

176 If the binding is for an SCA reference, the portType associated with the specified WSDL binding  
177 MUST be a compatible superset of the SCA reference interface as defined in the SCA Assembly  
178 Model specification [SCA-Assembly][SCA-Assembly], and the WSDL binding MUST satisfy all  
179 the policy constraints of the binding. If the binding is for an SCA reference, the portType  
180 associated with the specified WSDL binding MUST be a compatible superset of the SCA  
181 reference interface as defined in the SCA Assembly Model specification [SCA-Assembly], and  
182 the WSDL binding MUST satisfy all the policy constraints of the binding. [BWS20013] The SCA  
183 runtime MUST use the specified WSDL binding for invocations made using the SCA reference  
184 binding, or raise an error if it does not support the WSDL binding. The SCA runtime MUST use the  
185 specified WSDL binding for invocations made using the SCA reference binding, or raise an error  
186 if it does not support the WSDL binding. [BWS20014]

Formatted: Highlight

Formatted: Default Paragraph Font

187 When the *wsdl.binding* form of *wsdlElement* is used, the endpoint address URI for an SCA  
188 reference MUST be specified by either the *@uri* attribute on the binding or a WS-Addressing  
189 *EndpointReference* element, except where the SCA Assembly Model specification [SCA-  
190 *Assembly*][SCA-Assembly] states that the *@uri* attribute can be omitted. When the *wsdl.binding*  
191 form of *wsdlElement* is used, the endpoint address URI for an SCA reference MUST be specified  
192 by either the *@uri* attribute on the binding or a WS-Addressing *EndpointReference* element,  
193 except where the SCA Assembly Model specification [SCA-Assembly] states that the *@uri*  
194 attribute can be omitted. [BWS20015]

Formatted: Highlight

Formatted: Default Paragraph Font

Formatted: Font: Italic

Formatted: Font: Not Italic

195 • **/binding.ws/@wsdl:wsdlLocation** – when present this attribute specifies the location(s) of the  
196 WSDL document(s) associated with specific namespace(s).

197 The *@wsdl:wsdlLocation* attribute can be used in the event that the *<WSDL-namespace-URI>* value  
198 in the *@wsdlElement* attribute is not dereferencable, or when the intended WSDL document is to be  
199 found at a different location than the one pointed to by the *<WSDL-namespace-URI>*. The semantics  
200 of this attribute are specified in Section 7.1 of WSDL 2.0 [WSDL20].

201 If the *@wsdl:wsdlLocation* attribute is used the *@wsdlElement* attribute MUST also be specified. If the  
202 *@wsdl:wsdlLocation* attribute is used the *@wsdlElement* attribute MUST also be specified.  
203 [BWS20017]

Formatted: Font color: Auto

204 | **The value of the @wsdl:wsdlLocation attribute MUST identify an existing WSDL 1.1 document.**  
205 | **The value of the @wsdl:wsdlLocation attribute MUST identify an existing WSDL 1.1 document.**  
206 | **[BWS20018]**

Formatted: Font color: Auto

- 207 • **/binding.ws/wireFormat** – as defined in the SCA Assembly Specification **[SCA-Assembly]**. This  
208 specification does not define any new wireFormat elements.
- 209 • **/binding.ws/operationSelector** – as defined in the SCA Assembly Specification **[SCA-Assembly]**.  
210 This specification does not define any new operationSelector elements.
- 211 • **/binding.ws/endpointReference** – when present this element provides the WS-Addressing **[WS-**  
212 **Addr]** EndpointReference that specifies the endpoint for the service or reference.

213 | **A binding.ws element MUST NOT contain more than one of any of the following: the @uri attribute; the**  
214 | **@wsdlElement attribute referring to a WSDL port or to a WSDL service; the endpointReference**  
215 | **element.** **A binding.ws element MUST NOT contain more than one of any of the following: the @uri**  
216 | **attribute; the @wsdlElement attribute referring to a WSDL port or to a WSDL service; the**  
217 | **endpointReference element.** **[BWS20019]**

Formatted: Font color: Auto

218 | The endpoint address URI for an SCA service or the callback element of an SCA reference is determined  
219 | as specified in section 2.2. **For the callback element of an SCA service, the binding MUST NOT specify**  
220 | **an endpoint address URI or a WS-Addressing EndpointReference.** **For the callback element of an SCA**  
221 | **service, the binding MUST NOT specify an endpoint address URI or a WS-Addressing**  
222 | **EndpointReference.** **[BWS20020]**

223 | **The SCA runtime MUST support all the attributes of the <binding.ws> element, namely @name, @uri,**  
224 | **@requires, @policySets, @wsdlElement, and @wsdl:wsdlLocation.** **[BWS20021]**

225 | **The SCA runtime SHOULD support the element <endpointReference>.** **[BWS20022]** **If an SCA runtime**  
226 | **does not support the element <endpointReference>, then it MUST reject an SCA WS Binding XML**  
227 | **document (as defined in Section 5.1) that contains the element.** **[BWS20023]**

228 | **The <binding.ws> element MUST conform to the XML schema defined in sca-binding-webservice-**  
229 | **1.1.xsd.** **The <binding.ws> element MUST conform to the XML schema defined in sca-binding-**  
230 | **webservice-1.1.xsd.** **[BWS20024]**

## 231 | 2.1 Compatibility of SCA Service Interfaces and WSDL portTypes

232 | A WSDL portType is compatible with an SCA service interface if and only if all of these conditions are  
233 | satisfied:

- 234 | 1. The SCA service interface is remotable.
- 235 | 2. The operations on the portType are the same as the operations on the SCA service interface, with the  
236 | same operation name, same input types (taking order as significant), same output types (taking order  
237 | as significant), and same fault/exception types. If the SCA service interface is not a WSDL portType,  
238 | it is mapped to a WSDL portType for the purposes of this comparison. The mapping is defined in the  
239 | relevant SCA specification for the interface type. If the interface cannot be mapped to WSDL, the  
240 | SCA service interface is not compatible with the WSDL portType.
- 241 | 3. WSDL 1.1 message parts can point either to an XML Schema element declaration or to an XML  
242 | Schema type declaration. When determining compatibility between two WSDL operations, a  
243 | message part that points to an XML Schema element is considered to be incompatible with a  
244 | message part that points to an XML Schema type.
- 245 | 4. If either the portType or the SCA service interface declares an SCA callback interface, then both the  
246 | portType and the SCA service interface declare callback interfaces and these callback interfaces are  
247 | compatible according to points 1 through 3 above.

## 248 | 2.2 Endpoint URI resolution

249 | This specification does not mandate any particular way to determine the URI for a web services binding  
250 | on an SCA service. An absolute URI can be indicated by the @uri attribute, by the URI in a wsa:Address  
251 | element within an endpointReference element, or by the URI indicated in a WSDL port via a  
252 | @wsdlElement attribute. Implementations can use the specified URI as the service endpoint URI or they

253 can use a different URI which might include portions of the specified URI. For example, the service  
254 endpoint URI might be produced by modifying any or all of the host name, the port number, and a portion  
255 of the path.

256 Note that if no absolute URI is indicated by any of these elements, implementations can use the structural  
257 URI for the binding as a portion of the URI for the eventual deployed endpoint. In addition, the @uri  
258 attribute value could be relative; implementations are encouraged to combine this value with the structural  
259 URI for the service in determining a deployed URI.

260 The target address for a reference binding is defined as one of:

- 261 A. The value of the @uri attribute
- 262 B. The value of the wsa:Address element of the endpointReference element
- 263 C. The value of the address element of the WSDL port referenced by the @wsdlElement attribute
- 264 D. The value of the address element of one of the set of available WSDL ports as specified under the  
265 definition of the @wsdlElement attribute when it references a WSDL service element

266 If there is no target address for a reference binding, the SCA runtime MUST raise an error. [BWS20025]

267 For a reference binding, the SCA runtime MUST use the target address. For a reference binding, the SCA  
268 runtime MUST use the target address. [BWS20026]

## 269 2.3 Interface mapping

270 When binding.ws is used on a service or reference with an interface that is not defined by interface.wsdl,  
271 the SCA runtime MUST derive a WSDL portType for the service or reference from the interface using the  
272 WSDL-mapping rules defined for that SCA interface type. When binding.ws is used on a service or  
273 reference with an interface that is not defined by interface.wsdl, the SCA runtime MUST derive a WSDL  
274 portType for the service or reference from the interface using the WSDL-mapping rules defined for that  
275 SCA interface type. [BWS20027]

276 An SCA runtime MUST raise an error if the interface on a service or reference element with a binding.ws  
277 element does not map to a WSDL portType. An SCA runtime MUST raise an error if the interface on a  
278 service or reference element with a binding.ws element does not map to a WSDL portType. [BWS20028]

279 For example, for interface.java, the mapping to a WSDL portType is as defined in the SCA Java Common  
280 Annotations and API Specification [SCA-JCAA].

281 binding.ws implementations can use appropriate standards, for example WS-I AP 1.0 [WSI-AP] or MTOM  
282 [MTOM], to map interface parameters to binary attachments transparently to the target component.

## 283 2.4 Production of WSDL description for an SCA service

284 Any service hosted by an SCA runtime with one or more web service bindings with HTTP endpoints  
285 SHOULD return a WSDL description of the service in response to an HTTP GET request with the "?wsdl"  
286 suffix added to that HTTP endpoint URL. Any service hosted by an SCA runtime with one or more web  
287 service bindings with HTTP endpoints SHOULD return a WSDL description of the service in response to  
288 an HTTP GET request with the "?wsdl" suffix added to that HTTP endpoint URL. [BWS20029]

289 If none of the web service bindings for an SCA service have HTTP endpoints, then the SCA runtime  
290 SHOULD provide some other means of obtaining the WSDL description of the service. If none of the web  
291 service bindings for an SCA service have HTTP endpoints, then the SCA runtime SHOULD provide some  
292 other means of obtaining the WSDL description of the service. [BWS20030] This can include out of band  
293 mechanisms, for example publication to a UDDI registry.

294 Refer to section 4 for a detailed definition of the rules that are used for generating the WSDL description  
295 of an SCA service with one or more web service bindings.

## 296 2.5 Additional binding configuration data

297 SCA runtime implementations can provide additional metadata that is associated with a web service  
298 binding. This is done by providing extension points in the schema; refer to Appendix A: Web Services  
299 XML Binding Schema for the locations of these extension points.

300 This can be used for example to enable JAX-WS [JAX-WS] handlers to be executed as part of the target  
301 component dispatch. The specification of such metadata is SCA runtime-specific and is outside of the  
302 scope of this document.

## 303 2.6 Web Service Binding and SOAP Intermediaries

304 The Web Service binding does not provide any direct or explicit support for SOAP  
305 intermediaries [SOAP].

## 306 2.7 Support for WSDL extensibility

307 When a binding.ws element uses the @wsdlElement attribute, the details of the binding are specified by  
308 the WSDL element referenced by the value of the attribute. Per the WSDL specification, WSDL allows for  
309 extensibility via elements as well as attributes, and it specifies rules for processing such elements. This  
310 specification does not constrain the use of such extensibility in WSDL and relies on the rules specified in  
311 the WSDL specification for processing such extended elements.

312 An SCA runtime MUST support the WSDL extensions defined in the namespace associated with the  
313 prefix "sca" (as defined in section 1.1). An SCA runtime MUST support the WSDL extensions defined in  
314 the namespace associated with the prefix "sca" (as defined in section 1.1). [BWS20032]

315 The SCA runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP  
316 [WSDL11][WSDL11], as identified by the WSDL element wsoap11:binding that has the @transport  
317 attribute with a value of "http://schemas.xmlsoap.org/soap/http". The SCA runtime MUST support the  
318 WSDL 1.1 binding extension for SOAP 1.1 over HTTP [WSDL11], as identified by the WSDL element  
319 wsoap11:binding that has the @transport attribute with a value of "http://schemas.xmlsoap.org/soap/http".  
320 [BWS20033]

321 The SCA runtime SHOULD support the WSDL 1.1 binding extension for SOAP 1.2 over HTTP [W11-  
322 SOAP12][W11-SOAP12], as identified by the WSDL element wsoap12:binding that has the @transport  
323 attribute with a value of "http://schemas.xmlsoap.org/soap/http". The SCA runtime SHOULD support the  
324 WSDL 1.1 binding extension for SOAP 1.2 over HTTP [W11-SOAP12], as identified by the WSDL  
325 element wsoap12:binding that has the @transport attribute with a value of  
326 "http://schemas.xmlsoap.org/soap/http". [BWS20034]

327 Because a WSDL document might contain extension elements that cannot be supported by the SCA  
328 runtime, when using the @wsdlElement form of binding.ws it is not possible to determine whether the  
329 binding is supported by the SCA runtime without parsing the referenced WSDL element and its  
330 dependent elements.

## 331 2.8 Intents listed in the bindingType

332 This specification places no requirements on the intents [SCA-Policy] that are listed as either  
333 @alwaysProvides or @mayProvides in the bindingType for binding.ws.

## 334 2.9 Intents and binding configuration

335 This binding mandates support for SOAP 1.1 and encourages SOAP 1.2 support. The <bindingType>  
336 element associated with this binding MUST include the SOAP.v1\_1 intent in its @mayProvides or  
337 @alwaysProvides attributes. The <bindingType> element associated with this binding MUST include the  
338 SOAP.v1\_1 intent in its @mayProvides or @alwaysProvides attributes. [BWS20035] The <bindingType>  
339 element associated with this binding SHOULD include the SOAP.v1\_2 intent in its @mayProvides  
340 attribute. The <bindingType> element associated with this binding SHOULD include the SOAP.v1\_2 intent  
341 in its @mayProvides attribute. [BWS20036] For more details on the <bindingType> element see [SCA-  
342 Policy].

343 The SCA runtime MUST raise an error if a web service binding is configured with a policy intent(s) that  
344 conflicts with the binding instance's configuration. The SCA runtime MUST raise an error if a web service  
345 binding is configured with a policy intent(s) that conflicts with the binding instance's configuration.  
346 [BWS20037]

Formatted: Highlight

Formatted: Default Paragraph Font

Formatted: Highlight

Formatted: Default Paragraph Font

347 For example, it is an error to use the SOAP policy intent in combination with a WSDL binding that does  
348 not use SOAP.

## 349 **2.10 Support for WS-Addressing**

350 This binding mandates support for WS-Addressing 1.0 Core [WS-Addr], SOAP Binding [WS-Addr-  
351 SOAP], and the WS-Addressing WS-Policy assertion [WS-Addr-Meta]. The SCA runtime MUST support  
352 WS-Addressing 1.0 Core [WS-Addr] and WS-Addressing 1.0 SOAP Binding [WS-Addr-  
353 SOAP]. [BWS20038] The SCA runtime MUST support the WS-Addressing 1.0 WS-Policy assertion  
354 specified in [WS-Addr-Meta]. [BWS20039]

Formatted: Heading 2,H2

Formatted: Highlight

Formatted: Highlight

Formatted: Font color: Red

Formatted: Highlight

Formatted: Font color: Red

355

## 3 Web Service Binding Examples

356 The following snippets show the `sca.composite` file for the `MyValueComposite` file containing the service  
357 element for the `MyValueService` and reference element for the `StockQuoteService`. Both the service and  
358 the reference use a Web Service binding.

### 3.1 Example Using WSDL documents

360 Snippet 3-1 shows a service and reference using the SCA Web Service binding, using existing WSDL  
361 documents in both cases. In each case there is a single binding element, whose name defaults to the  
362 service/reference name.

363 The service's binding is defined by the WSDL document associated with the given URI. This service  
364 conforms to WS-I Basic Profile 1.1.

365 The first reference's binding is defined by the specified WSDL service in the WSDL document at the given  
366 location. The reference can use any of the WSDL service's ports to invoke the target service. The  
367 second reference's binding is defined by the specified WSDL binding. The specific endpoint URI to be  
368 invoked is provided via the `@uri` attribute.

369

```
370 <?xml version="1.0" encoding="ASCII"?>
371 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
372   name="MyValueComposite">
373   <service name="MyValueService">
374     <interface.java interface="services.myvalue.MyValueService"/>
375     <binding.ws wsdlElement="http://www.example.org/MyValueService#
376       wsdl.binding(MyValueService/MyValueServiceSOAP)"/>
377     ...
378   </service>
379   ...
380   ...
381   <reference name="StockQuoteReference1">
382     <interface.java interface="services.stockquote.StockQuoteService"/>
383     <binding.ws wsdlElement="http://www.example.org/StockQuoteService#
384       wsdl.service(StockQuoteService) "
385       wsdl.binding(StockQuoteService) "
386     wsdl:wsdlLocation="http://www.example.org/StockQuoteService
387       http://www.example.org/StockQuoteService.wsdl"/>
388   </reference>
389   <reference name="StockQuoteReference2">
390     <interface.java interface="services.stockquote.StockQuoteService"/>
391     <binding.ws wsdlElement="http://www.example.org/StockQuoteService#
392       wsdl.binding(StockQuoteBinding) "
393     wsdl:wsdlLocation="http://www.example.org/StockQuoteService
394       http://www.example.org/StockQuoteService.wsdl"
395     uri="http://www.example.org/StockQuoteService5"/>
396   </reference>
397 </composite>
```

399 *Snippet 3-1: Example Binding with a WSDL Document*

### 3.2 Examples Without a WSDL Document

401 Snippet 3-2 shows the simplest form of the binding element without WSDL document, assuming all  
402 defaults for portType mapping and SOAP binding synthesis. The service and reference each have a  
403 single binding element, whose name defaults to the service/reference name.

404 The service is to be made available at a location determined by the deployment of this component. It will  
405 have a single port address and SOAP binding, with a simple WS-I BasicProfile 1.1 compliant binding, and  
406 using the default options for mapping the Java interface to a WSDL portType.

407 The reference indicates a service to be invoked which has a SOAP binding and portType that matches  
408 the default options for binding synthesis and interface mapping. One particular use of this case would be  
409 where the reference is to an SCA service with a web service binding which itself uses all the defaults.

410

```
411 <?xml version="1.0" encoding="ASCII"?>
412 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
413     name="MyValueComposite">
414
415     <service name="MyValueService">
416         <interface.java interface="services.myvalue.MyValueService"/>
417         <binding.ws/>
418         ...
419     </service>
420
421     ...
422
423     <reference name="StockQuoteService">
424         <interface.java interface="services.stockquote.StockQuoteService"/>
425         <binding.ws uri="http://www.example.org/StockQuoteService"/>
426     </reference>
427 </composite>
```

428 *Snippet 3-2: Example Binding without a WSDL Document*

429

430 Snippet 3-3 shows the use of the binding element without a WSDL document, with multiple SOAP  
431 bindings with non-default values. The SOAP 1.2 binding name defaults to the service name, the SOAP  
432 1.1 binding is given an explicit name. The reference has a web service binding which uses SOAP 1.2,  
433 but otherwise uses all the defaults for SOAP binding. The reference binding name defaults to the  
434 reference name.

435

```
436 <?xml version="1.0" encoding="ASCII"?>
437 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
438     name="MyValueComposite">
439
440     <service name="MyValueService">
441         <interface.java interface="services.myvalue.MyValueService"/>
442         <binding.ws name="MyValueServiceSOAP11" requires="SOAP.v1_1"/>
443         <binding.ws requires="SOAP.v1_2"/>
444         ...
445     </service>
446
447     ...
448
449     <reference name="StockQuoteService">
450         <interface.java interface="services.stockquote.StockQuoteService"/>
451         <binding.ws uri="http://www.example.org/StockQuoteService"
452             requires="SOAP.v1_2"/>
453     </reference>
454 </composite>
```

455 *Snippet 3-3: Example Binding with Multiple SOAP Bindings*



## 456 4 Transport Binding

457 The binding.ws element provides numerous ways to specify exactly how messages ought to be  
458 transmitted from or to the reference or service. Those ways include references to WSDL binding elements  
459 from the @wsdlElement attribute, policy intents, and even vendor extensions within the binding.ws  
460 element. This section describes the defaults to be used if the specific transport details are not otherwise  
461 specified.

### 462 4.1 Intents

463 So as to narrow the range of choices for how messages are carried, these policy intents affect the  
464 transport binding:

- 465 • SOAP

466 **When the SOAP intent is required, the SCA runtime MUST transmit and receive messages using**  
467 **SOAP. One or more SOAP versions can be used. When the SOAP intent is required, the SCA runtime**  
468 **MUST transmit and receive messages using SOAP. One or more SOAP versions can be used.**  
469 **[BWS40001]**

- 470 • SOAP.v1\_1

471 **When the SOAP.v1\_1 intent is required, the SCA runtime MUST transmit and receive messages**  
472 **using only SOAP 1.1. When the SOAP.v1\_1 intent is required, the SCA runtime MUST transmit and**  
473 **receive messages using only SOAP 1.1. [BWS40002]**

- 474 • SOAP.v1\_2

475 **When the SOAP.v1\_2 intent is required, the SCA runtime MUST transmit and receive messages**  
476 **using only SOAP 1.2. When the SOAP.v1\_2 intent is required, the SCA runtime MUST transmit and**  
477 **receive messages using only SOAP 1.2. [BWS40003]**

### 478 4.2 Default Transport Binding Rules

#### 479 4.2.1 WS-I Basic Profile Alignment

480 To align to WS-I Basic Profile, the resulting WSDL port needs to be all document-literal, or all rpc-literal  
481 binding (per WS-I Basic Profile 1.1 R2705 **[WSI-Profiles]**). This means, for any given portType, for all  
482 messages referenced by all operations in that portType, either

- 483 • that every message part references an XML Schema type (rpc-literal pattern)
- 484 • or that every message references exactly zero or one XML Schema elements (document-literal  
485 pattern)

486 **For an SCA service or reference element, the portType from the service's or reference's interface or**  
487 **derived from that interface MUST follow either the rpc-literal pattern or the document-literal pattern. For an**  
488 **SCA service or reference element, the portType from the service's or reference's interface or derived from**  
489 **that interface MUST follow either the rpc-literal pattern or the document-literal pattern. [BWS40004]**

490 The rest of this section assumes the short-hand reference of a "rpc-literal" or "document-literal" pattern,  
491 depending on which of the two bullet points above it matches.

#### 492 4.2.2 Default Transport Binding Rules

493 The **default transport binding rules** for the Web Service binding are:

- 494 • HTTP-based transfer protocol;
- 495 • SOAP 1.1 binding;
- 496 • "literal" format as described in section 3.5 of **[WSDL11]**;

- 497 • Either the document literal or rpc literal pattern, depending on the service or reference interface as  
498 described in section 4.2.1;
- 499 – For document literal pattern, each message uses "document" style, as per section 3.5 of  
500 **[WSDL11]**;
- 501 – For rpc-literal pattern, each message uses "rpc" style, as per section 3.5 of **[WSDL11]** and the  
502 child elements of the SOAP Body element are namespace qualified with a non-empty namespace  
503 name;
- 504 • For SOAP 1.1 messages, the SOAPAction HTTP header described in section 6.1.1 of **[SOAP11]**  
505 represents the empty string, in quotes ("");
- 506 • For SOAP 1.2 messages, the SOAP Action feature described in section 6.5 of **[SOAP12Adjuncts]**  
507 does not appear;
- 508 • All WSDL message parts are carried in the SOAP body.

509 **In the event that the transport details are not determined by use of the @wsdlElement attribute, @uri**  
510 **attribute, endpointReference element, policy intents, policy sets or extensions to the binding.ws element,**  
511 **an SCA runtime MUST enable the default transport binding rules. In the event that the transport details**  
512 **are not determined by use of the @wsdlElement attribute, @uri attribute, endpointReference element,**  
513 **policy intents, policy sets or extensions to the binding.ws element, an SCA runtime MUST enable the**  
514 **default transport binding rules. [BWS40005]**

515 When using the default transport binding rules, the SCA runtime can provide additional WSDL bindings,  
516 unless policy is applied that explicitly restricts this.

517 **When using the default transport binding rules with the rpc-literal pattern, the SCA runtime SHOULD use**  
518 **the structural URI associated with the binding as the namespace of the child elements of the SOAP body**  
519 **element. When using the default transport binding rules with the rpc-literal pattern, the SCA runtime**  
520 **SHOULD use the structural URI associated with the binding as the namespace of the child elements of**  
521 **the SOAP body element. [BWS40007]**

522

## 5 Implementing SCA Callbacks using Web Services

523

### 5.1 SCA Web Services Callback Protocol

524

This section defines a SOAP- and WS-Addressing-based SCA Web Services callback protocol that can be used to implement a bidirectional interface [SCA-Assembly]. For examples of wire messages exchanged when using this protocol see Appendix E.

527

The protocol involves two communicating parties: a Service that implements the SCA bidirectional interface using Web services (WSCB Service) and a client that invokes the SCA bidirectional interface using Web services (WSCB Client). The WSCB Service implements the forward interface and the WSCB Client implements the callback interface. SCA Web Services Callback Protocol involves the following rules.

532

1. Every request message from the WSCB Client that invokes the forward interface MUST contain a Callback EPR. Every request message from the WSCB Client that invokes the forward interface MUST contain a Callback EPR. [BWS50002] If the request message contains the wsa:From SOAP header block then the wsa:From header block specifies the Callback EPR. If the wsa:From header block is not present then the wsa:ReplyTo header block specifies the Callback EPR.

537

If the Callback EPR's [address] value is

538

"http://www.w3.org/2005/08/addressing/anonymous" or

539

"http://www.w3.org/2005/08/addressing/none" then the WSCB Service MUST generate

540

the Invalid Addressing Header fault as specified in Section 6.4.1 of [WS-Addr-SOAP][WS-Addr-

541

SOAP]. If the Callback EPR's [address] value is

542

"http://www.w3.org/2005/08/addressing/anonymous" or

543

"http://www.w3.org/2005/08/addressing/none" then the WSCB Service MUST generate

544

the Invalid Addressing Header fault as specified in Section 6.4.1 of [WS-Addr-SOAP]. [BWS50004]

545

Such a fault can include additional [Subsubcode]

546

wsa:OnlyNonAnonymousAddressSupported.

547

2. A request message that invokes the forward interface can contain the wsa:MessageID SOAP header block. If there is a need to have the callback request message correlated to an individual forward request message, the wsa:MessageID SOAP header block can be used for this purpose.

550

3. When the WSCB Service invokes the callback interface, it MUST use the Callback EPR from a request message that invoked the forward interface. When the WSCB Service invokes the callback interface, it MUST use the Callback EPR from a request message that invoked the forward interface. [BWS50005] Once the Callback EPR is selected, the WSCB Service MUST follow the rules defined in Section 3.3 of [WS-Addr][WS-Addr] to invoke operations on the callback interface. Once the Callback EPR is selected, the WSCB Service MUST follow the rules defined in Section 3.3 of [WS-Addr] to invoke operations on the callback interface. [BWS50006]

557

When the WSCB Service invokes the callback interface, if the request message from which the Callback EPR was obtained contained the wsa:MessageID SOAP header block, the WSCB Service MUST include a wsa:RelatesTo SOAP header block in the callback message. When the WSCB Service invokes the callback interface, if the request message from which the Callback EPR was obtained contained the wsa:MessageID SOAP header block, the WSCB Service MUST include a wsa:RelatesTo SOAP header block in the callback message. [BWS50007] The wsa:RelatesTo SOAP header block MUST have the relationship type value of "http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback" and the related message id MUST be the wsa:MessageID of the message from which the Callback EPR was obtained. The wsa:RelatesTo SOAP header block MUST have the relationship type value of "http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback" and the related message id MUST be the wsa:MessageID of the message from which the Callback EPR was obtained. [BWS50008]

568

Formatted: Highlight

Formatted: Default Paragraph Font

Formatted: Highlight

Formatted: Font: (Default) Courier New

Formatted: Font: (Default) Arial

Formatted: Font: Courier New

Formatted: Font: Arial

569 If the request message from which the Callback EPR was obtained did not contain the `wsa:MessageID`  
570 SOAP header block, the WSCB Service MUST NOT include a `wsa:RelatesTo` SOAP header block with  
571 a relationship type value of "`http://docs.oasis-open.org/opencsa/sca-`  
572 `bindings/ws/callback`" in the callback message. If the request message from which the Callback  
573 EPR was obtained did not contain the `wsa:MessageID` SOAP header block, the WSCB Service MUST  
574 NOT include a `wsa:RelatesTo` SOAP header block with a relationship type value of  
575 "`http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback`" in the callback  
576 message. [BWS50009]

577 When a service that offers a bidirectional interface is invoked, depending on the semantics and/or  
578 implementation of the service, it is possible that the service might invoke the callback interface before the  
579 forward operation ends. In such cases, it is necessary for the binding on the reference-side to be listening  
580 for callback request(s) from the service, before the forward operation request is sent on the wire to the  
581 service, and continue listening as long as callback requests are expected. It is possible that before the  
582 response to the forward request is sent a response to one or more callback requests are required by the  
583 service.

## 584 5.2 SCA Web Services Callback Protocol with WS-MakeConnection

585 It is possible that the invoker of a service that uses a bidirectional interface has a binding that cannot  
586 accept connections for callbacks from a service (for example, when it has the `noListener` intent [**SCA-**  
587 **Policy**]). When this is the case, it is necessary for the binding to support a polling mechanism. An  
588 example of a polling mechanism is WS-MakeConnection [**WS-MC**]. This section describes the use of the  
589 SCA Web Services Callback Protocol in conjunction with WS-MakeConnection. For examples of wire  
590 messages exchanged when using the SCA Web Services Callback protocol in conjunction with WS-  
591 MakeConnection see Appendix E.1.

592 When the SCA Web Services Callback protocol is implemented in conjunction with WS-MakeConnection,  
593 it has to adhere to the rules described for the SCA Web Services Callback Protocol and also to those of  
594 WS-MakeConnection.

595 The Callback EPR's [address] value present in the request message that invoked the forward interface  
596 follows the form of the MakeConnection Anonymous URI, i.e. "`http://docs.oasis-open.org/ws-`  
597 `rx/wsmc/200702/anonymous?id={unique-String}`".

598 The `unique-String` value is a globally unique value such as a UUID, as defined by the WS-  
599 MakeConnection specification.

600 When the service implementation invokes the callback interface, it uses the Callback EPR from a request  
601 message that invoked the forward interface, and the callback request message is sent as the response to  
602 a `wsmc:MakeConnection` message that contains the `wsmc:Address` value that matches the  
603 MakeConnection Anonymous URI in the Callback EPR.

604 When a service that offers a bidirectional interface is invoked using WS-MakeConnection Anonymous  
605 URI as the value for the Callback EPR address, depending on the semantics and/or implementation of  
606 the service, it is possible that the service might invoke the callback interface before the forward operation  
607 ends. In such cases, it is necessary for the binding on the reference-side to start polling for callback  
608 request(s) from the service, before or right after the forward operation request is sent and before a  
609 response is received, and continue polling as long as callback requests are expected. It is possible that  
610 before the response to the forward request is sent a response to one or more callback requests are  
611 required by the service.

## 612 5.3 Policy Assertion for SCA Web Services Callback Protocol

613 WS-Policy Framework [**WS-Policy**] and WS-Policy Attachment [**WS-PA**] collectively define a framework,  
614 model and grammar for expressing the requirements, and general characteristics of entities in an XML  
615 Web services-based system. To enable a Web service client and a Web service to describe their  
616 requirements for implementing SCA Web Services Callback Protocol, this specification defines a single  
617 policy assertion that leverages the WS-Policy framework.

### 618 5.3.1 Assertion Model

619 The WSCallback policy assertion indicates that the WSCB Client and the WSCB Service MUST use SCA  
620 Web Services Callback Protocol to implement callbacks. The WSCallback policy assertion indicates that  
621 the WSCB Client and the WSCB Service MUST use SCA Web Services Callback Protocol to implement  
622 callbacks. [BWS50010] Specifically, the protocol determines the requirements on the forward request  
623 message, the EPR used for callbacks and the requirements on the callback request message.

### 624 5.3.2 Normative Outline

625 The normative outline for the WSCallback assertion is:

626

```
627 <sca:WSCallback ...>  
628   ...  
629 </sca:WSCallback>
```

630 *Snippet 5-1: WSCallback Assertion*

631

632 The content model of the WSCallback element is:

- 633 • /sca:WSCallback: A policy assertion that specifies that SCA Web Services Callback protocol is  
634 used when sending messages.

### 635 5.3.3 Assertion Attachment

636 The WSCallback policy assertion can have the following Policy Subjects [WS-PA]:

- 637 • Endpoint Policy Subject

638 WS-PolicyAttachment defines a set of WSDL/1.1 policy attachment points for each of the above Policy  
639 Subjects. Since a WSCallback policy assertion specifies a concrete behavior, it cannot be attached to the  
640 abstract WSDL policy attachment points.

641 The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects allowed for a  
642 WSCallback policy assertion but which MUST NOT have WSCallback policy assertions attached:  
643 wsdl:portType The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects  
644 allowed for a WSCallback policy assertion but which MUST NOT have WSCallback policy assertions  
645 attached: wsdl:portType [BWS50013]

646 The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects allowed for a  
647 WSCallback policy assertion and which can have WSCallback policy assertions attached:

- 648 • wsdl:port
- 649 • wsdl:binding

### 650 5.3.4 Assertion Example

651 Snippet 5-2 the use of the WSCallback policy assertion in a WSDL document.

652

```
653 (01) <wsdl:definitions  
654 (02)   targetNamespace="example.com"  
655 (03)   xmlns:tns="example.com"  
656 (04)   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"  
657 (05)   xmlns:wsp="http://www.w3.org/ns/ws-policy"  
658 (06)   xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
659 (07)   xmlns:wssu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
660 wsssecurity-utility-1.0.xsd">  
661 (08)  
662 (09) <wsp:UsingPolicy wsdl:required="true" />  
663 (10)
```

```
664 (11) <wsp:Policy wsu:Id="MyPolicy" >
665 (12)   <sca:WSCallback/>
666 (13) </wsp:Policy>
667 (14)
668 (15) <!-- omitted elements -->
669 (16)
670 (17) <wsdl:binding name="MyBinding" type="tns:MyPortType" >
671 (18)   <wsp:PolicyReference URI="#MyPolicy" />
672 (19)   <!-- omitted elements -->
673 (20) </wsdl:binding>
674 (21)
675 (22)</wsdl:definitions>
```

676 *Snippet 5-2: WSCallback Policy Assertion Used in a WSDL Document*

677

678 Line (09) in Snippet 5-2 indicates that WS-Policy is in use as a required extension. Lines (11-13) are a  
679 policy expression that includes a WSCallback policy assertion (line 12) to indicate that SCA Web Services  
680 Callback protocol is used. Lines (17-20) are a WSDL binding. Line (18) indicates that the policy in lines  
681 (11-13) applies to this binding, specifically indicating that SCA Web Services Callback protocol is used  
682 over all the messages in the binding.

### 683 5.3.5 Security Considerations

684 Policies and assertions SHOULD be signed to prevent tampering. [BWS50014] Policies SHOULD NOT  
685 be accepted unless they are signed and have an associated security token to specify the signer has  
686 proper claims for the given policy. [BWS50015] That is, a relying party shouldn't rely on a policy unless  
687 the policy is signed and presented with sufficient claims to pass the relying parties acceptance criteria.

688 Note that the mechanisms described in this document could be secured as part of a SOAP message  
689 using WS-Security [WS-Security] or embedded within other objects using object-specific security  
690 mechanisms.

---

## 691 6 Conformance

692 The XML schema pointed to by the RDDL document at the namespace URI, defined by this specification,  
693 are considered to be authoritative and take precedence over the XML schema defined in the appendix of  
694 this document.

695 This specification defines four targets for conformance:

- 696 a) SCA WS Binding XML Document
- 697 b) Web Service Callback Service (WSCB Service)
- 698 c) Web Service Callback Client (WSCB Client)
- 699 d) SCA Runtime

### 700 6.1 SCA WS Binding XML Document

701 An SCA WS Binding XML document is an SCA Composite Document, or an SCA ComponentType  
702 Document, as defined by the SCA Assembly specification Section 13.1 **[SCA-Assembly]**, that uses the  
703 <binding.ws> element.

704 An SCA WS Binding XML document **MUST** be a conformant SCA Composite Document or a SCA  
705 ComponentType Document, as defined by the SCA Assembly specification **[SCA-Assembly]**, and **MUST**  
706 comply with all statements in Appendix C: Conformance Items related to elements and attributes in an  
707 SCA WS Binding XML document, notably all "MUST" statements have to be implemented.

### 708 6.2 Web Service Callback Service

709 An implementation that claims to conform to the requirements of a WSCB Service defined in this  
710 specification **MUST** conform to all the statements in Appendix C: Conformance Items related to a WSCB  
711 Service.

712

### 713 6.3 Web Service Callback Client

714 An implementation that claims to conform to the requirements of a WSCB Client defined in this  
715 specification **MUST** conform to all the statements in Appendix C: Conformance Items related to a WSCB  
716 Client.

717

### 718 6.4 SCA Runtime

719 An implementation that claims to conform to the requirements of an SCA Runtime defined in this  
720 specification has to meet the following conditions:

- 721 1. The implementation **MUST** comply with all statements in Appendix C: Conformance Items related to  
722 an SCA Runtime, except for those that originate from Section 5, notably all "MUST" statements have  
723 to be implemented.
- 724 2. The implementation **MAY** support the SCA Web Services Callback Protocol. If it does, it **MUST** be a  
725 compliant WSCB Service and WSCB Client.
- 726 3. The implementation **MAY** support the SCA Web Services Callback Protocol in conjunction with WS-  
727 MakeConnection. If it does, it **MUST** be a compliant WSCB Service, WSCB Client, and it **MUST**  
728 comply with the requirements of WS-MakeConnection and **MUST** support the [MakeConnection policy](#)  
729 **assertion**.
- 730 4. The implementation **MUST** conform to the SCA Assembly Model Specification Version 1.1 **[SCA-**  
731 **Assembly]**, and to the SCA Policy Framework Version 1.1 **[SCA-Policy]**.

732 5. The implementation MUST reject a SCA WS Binding XML Document that is not conformant per  
733 Section 6.1.

734 Note that when an SCA Runtime implementation claims to conform to the SCA Web Services Callback  
735 Protocol, the implementation acts as a WSCB Service/Client on behalf of an SCA component. In such a  
736 case the component developer does not have to implement the protocol and can rely on the SCA  
737 Runtime's support of the protocol.



738  
739

## A. Web Services XML Binding Schema: sca-binding- ws-1.1.xsd (Normative)

740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (C) OASIS (R) 2005,2010. All Rights Reserved.
      OASIS trademark, IPR and other policies apply. -->
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://docs.oasis-open.org/ns/opencsa/sca/200912"
  xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200912"
  xmlns:wsdli="http://www.w3.org/ns/wsdli-instance"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  elementFormDefault="qualified">

  <import namespace="http://www.w3.org/ns/wsdli-instance"
    schemaLocation="http://www.w3.org/2007/05/wsdli/wsdli20-
instance.xsd"/>
  <import namespace="http://www.w3.org/2005/08/addressing"
    schemaLocation="http://www.w3.org/2006/03/addressing/ws-
addr.xsd"/>

  <include schemaLocation="sca-core-1.1-cd05.xsd"/>

  <element name="binding.ws" type="sca:WebServiceBinding"
    substitutionGroup="sca:binding"/>

  <complexType name="WebServiceBinding">
    <complexContent>
      <extension base="sca:Binding">
        <sequence>
          <element name="endpointReference"
type="wsa:EndpointReferenceType"
            minOccurs="0" maxOccurs="unbounded"/>
          <any namespace="##other" processContents="lax"
            minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="wsdlElement" type="anyURI" use="optional"/>
        <attribute ref="wsdli:wsdliLocation" use="optional"/>
      </extension>
    </complexContent>
  </complexType>
</schema>
```

778  
779  
780

## B. SCA Web Services Callback Protocol Policy Assertion XML Schema: sca-binding-webservice- callback-1.1.xsd (Normative)

781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- (c) Copyright OASIS 2005, 2010. All Rights Reserved.
      OASIS trademark, IPR and other policies apply. -->

<schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://docs.oasis-open.org/ns/opencsa/sca/200912"
  elementFormDefault="qualified">

  <element name="WSCallback">
    <complexType>
      <sequence>
        <any namespace="##other" processContents="lax" minOccurs="0"
          maxOccurs="unbounded"/>
      </sequence>
      <anyAttribute namespace="##any" processContents="lax"/>
    </complexType>
  </element>

</schema>
```

800

## C. Conformance Items (Normative)

801

This section contains a list of conformance items for the SCA Web Service Binding specification.

Conformance ID	Description
[BWS20001][BWS20004]	For an SCA reference, the @uri attribute MUST be an absolute value.
[BWS20002]	The value of the @wsdlElement attribute MUST identify an element in an existing WSDL 1.1 document.
[BWS20003]	If the binding is for an SCA service, the wsdlElement attribute MUST NOT specify the wsdl.service form of URI.
[BWS20004][BWS20004]	If the wsdl.service form of wsdlElement is used on an SCA reference binding, the set of available ports for that reference binding MUST be non-empty.
[BWS20005]	If the wsdl.service form of wsdlElement is used on an SCA reference binding, the SCA runtime MUST raise an error if there are no available ports that it supports.
[BWS20006][BWS20006]	When an invocation is made using an SCA reference binding with the wsdl.service form of wsdlElement, the SCA runtime MUST use exactly one port from the set of available ports for the reference (with port selection on a per-invocation basis permitted).
[BWS20007][BWS20007]	If the binding is for an SCA service, the portType associated with the specified WSDL port MUST be compatible with the SCA service interface as defined in section 2.1, and the port MUST satisfy all the policy constraints of the binding.
[BWS20008]	The SCA runtime MUST expose an endpoint for the specified WSDL port, or raise an error if it does not support the WSDL port.
[BWS20009]	If the binding is for an SCA reference, the portType associated with the specified WSDL port MUST be a compatible superset of the SCA reference interface as defined in the SCA Assembly Model specification [SCA-Assembly][SCA-Assembly], and the port MUST satisfy all the policy constraints of the binding.
[BWS20010]	The SCA runtime MUST use the specified WSDL port for invocations made using the SCA reference binding, or raise an error if it does not support the WSDL port.
[BWS20011]	If the binding is for an SCA service, the portType associated with the specified WSDL binding MUST be compatible with the SCA service interface as defined in section 2.1, and the WSDL binding MUST satisfy all the policy constraints of the binding.
[BWS20012]	The SCA runtime MUST expose an endpoint for the specified WSDL binding, or raise an error if it does not support the WSDL binding.
[BWS20013]	If the binding is for an SCA reference, the portType associated with the specified WSDL binding MUST be a compatible superset of the SCA reference interface as defined in the SCA Assembly Model specification [SCA-Assembly][SCA-Assembly], and the WSDL binding MUST satisfy all the policy constraints of the binding.

Formatted: Highlight

Formatted: Highlight

[BWS20014]	The SCA runtime MUST use the specified WSDL binding for invocations made using the SCA reference binding, or raise an error if it does not support the WSDL binding.
[BWS20015]	When the <i>wSDL.binding</i> form of <i>wSDL.Element</i> is used, the endpoint address URI for an SCA reference MUST be specified by either the <i>@uri</i> attribute on the binding or a WS-Addressing <i>EndpointReference</i> element, except where the SCA Assembly Model specification [SCA-Assembly][SCA-Assembly] states that the <i>@uri</i> attribute can be omitted.
[BWS20017]	If the <i>@wsdl:wsdlLocation</i> attribute is used the <i>@wsdlElement</i> attribute MUST also be specified.
[BWS20018]	The value of the <i>@wsdl:wsdlLocation</i> attribute MUST identify an existing WSDL 1.1 document.
[BWS20019]	A <i>binding.ws</i> element MUST NOT contain more than one of any of the following: the <i>@uri</i> attribute; the <i>@wsdlElement</i> attribute referring to a WSDL port or to a WSDL service; the <i>endpointReference</i> element.
[BWS20020]	For the <i>callback</i> element of an SCA service, the binding MUST NOT specify an endpoint address URI or a WS-Addressing <i>EndpointReference</i> .
[BWS20021]	The SCA runtime MUST support all the attributes of the <i>&lt;binding.ws&gt;</i> element, namely <i>@name</i> , <i>@uri</i> , <i>@requires</i> , <i>@policySets</i> , <i>@wsdlElement</i> , and <i>@wsdl:wsdlLocation</i> .
[BWS20022]	The SCA runtime SHOULD support the element <i>&lt;endpointReference&gt;</i> .
[BWS20023]	If an SCA runtime does not support the element <i>&lt;endpointReference&gt;</i> , then it MUST reject an SCA WS Binding XML document (as defined in Section 5.1) that contains the element.
[BWS20024]	The <i>&lt;binding.ws&gt;</i> element MUST conform to the XML schema defined in <i>sca-binding-webservice-1.1.xsd</i> .
[BWS20025]	If there is no target address for a reference binding, the SCA runtime MUST raise an error.
[BWS20026]	For a reference binding, the SCA runtime MUST use the target address.
[BWS20027]	When <i>binding.ws</i> is used on a service or reference with an interface that is not defined by <i>interface.wSDL</i> , the SCA runtime MUST derive a WSDL <i>portType</i> for the service or reference from the interface using the WSDL-mapping rules defined for that SCA interface type.
[BWS20028]	An SCA runtime MUST raise an error if the interface on a service or reference element with a <i>binding.ws</i> element does not map to a WSDL <i>portType</i> .
[BWS20029]	Any service hosted by an SCA runtime with one or more web service bindings with HTTP endpoints SHOULD return a WSDL description of the service in response to an HTTP GET request with the "?wsdl" suffix added to that HTTP endpoint URL.
[BWS20030]	If none of the web service bindings for an SCA service have HTTP endpoints, then the SCA runtime SHOULD provide some other means of obtaining the WSDL description of the service.
[BWS20032]	An SCA runtime MUST support the WSDL extensions defined in the namespace associated with the prefix "sca" (as defined in section 1.1).

Formatted: Highlight

[BWS20033]	The SCA runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP <b>[WSDL11][WSDL11]</b> , as identified by the WSDL element wsoap11:binding that has the @transport attribute with a value of "http://schemas.xmlsoap.org/soap/http".
[BWS20034]	The SCA runtime SHOULD support the WSDL 1.1 binding extension for SOAP 1.2 over HTTP <b>[W11-SOAP12][W11-SOAP12]</b> , as identified by the WSDL element wsoap12:binding that has the @transport attribute with a value of "http://schemas.xmlsoap.org/soap/http".
[BWS20035]	The <bindingType> element associated with this binding MUST include the SOAP.v1_1 intent in its @mayProvides or @alwaysProvides attributes.
[BWS20036]	The <bindingType> element associated with this binding SHOULD include the SOAP.v1_2 intent in its @mayProvides attribute.
[BWS20037]	The SCA runtime MUST raise an error if a web service binding is configured with a policy intent(s) that conflicts with the binding instance's configuration.
[BWS20038]	The SCA runtime MUST support WS-Addressing 1.0 Core <b>[WS-Addr]</b> and WS-Addressing 1.0 SOAP Binding <b>[WS-Addr-SOAP]</b> .
[BWS20039]	The SCA runtime MUST support the WS-Addressing 1.0 WS-Policy assertion specified in <b>[WS-Addr-Meta]</b> .
[BWS40001]	When the SOAP intent is required, the SCA runtime MUST transmit and receive messages using SOAP. One or more SOAP versions can be used.
[BWS40002]	When the SOAP.v1_1 intent is required, the SCA runtime MUST transmit and receive messages using only SOAP 1.1.
[BWS40003]	When the SOAP.v1_2 intent is required, the SCA runtime MUST transmit and receive messages using only SOAP 1.2.
[BWS40004]	For an SCA service or reference element, the portType from the service's or reference's interface or derived from that interface MUST follow either the rpc-literal pattern or the document-literal pattern.
[BWS40005]	In the event that the transport details are not determined by use of the @wsdlElement attribute, @uri attribute, endpointReference element, policy intents, policy sets or extensions to the binding.ws element, an SCA runtime MUST enable the default transport binding rules.
[BWS40007]	When using the default transport binding rules with the rpc-literal pattern, the SCA runtime SHOULD use the structural URI associated with the binding as the namespace of the child elements of the SOAP body element.
[BWS50002]	Every request message from the WSCB Client that invokes the forward interface MUST contain a Callback EPR.
[BWS50004]	If the Callback EPR's [address] value is "http://www.w3.org/2005/08/addressing/anonymous" or "http://www.w3.org/2005/08/addressing/none" then the WSCB Service MUST generate the Invalid Addressing Header fault as specified in Section 6.4.1 of <b>[WS-Addr-SOAP][WS-Addr-SOAP]</b> .
[BWS50005]	When the WSCB Service invokes the callback interface, it MUST use the Callback EPR from a request message that invoked the forward interface.
[BWS50006]	Once the Callback EPR is selected, the WSCB Service MUST follow the rules defined in Section 3.3 of <b>[WS-Addr][WS-Addr]</b> to invoke operations on the

Formatted: Highlight

Formatted: Highlight

Formatted: Font color: Red

Field Code Changed

Formatted: Highlight

Field Code Changed

Formatted: Highlight

Formatted: Font color: Red

Field Code Changed

Formatted: Highlight

Formatted: Highlight

Formatted: Highlight

	callback interface.
[BWS50007]	When the WSCB Service invokes the callback interface, if the request message from which the Callback EPR was obtained contained the <code>wsa:MessageID</code> SOAP header block, the WSCB Service MUST include a <code>wsa:RelatesTo</code> SOAP header block in the callback message.
[BWS50008]	The <code>wsa:RelatesTo</code> SOAP header block MUST have the relationship type value of " <code>http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback</code> " and the related message id MUST be the <code>wsa:MessageID</code> of the message from which the Callback EPR was obtained.
[BWS50009][BWS50009]	If the request message from which the Callback EPR was obtained did not contain the <code>wsa:MessageID</code> SOAP header block, the WSCB Service MUST NOT include a <code>wsa:RelatesTo</code> SOAP header block with a relationship type value of " <code>http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback</code> " in the callback message.
[BWS50010]	The WSCallback policy assertion indicates that the WSCB Client and the WSCB Service MUST use SCA Web Services Callback Protocol to implement callbacks.
[BWS50013]	The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects allowed for a WSCallback policy assertion but which MUST NOT have WSCallback policy assertions attached: <code>wsdl:portType</code>
[BWS50014]	Policies and assertions SHOULD be signed to prevent tampering.
[BWS50015]	Policies SHOULD NOT be accepted unless they are signed and have an associated security token to specify the signer has proper claims for the given policy.

---

## 802 D. WSDL Generation (Non-Normative)

803 Due to the number of factors that determine how a WSDL might be generated, including compatibility with  
804 existing WSDL uses, precise details cannot be specified. For example, implementation decisions can  
805 affect the way WSDL might be generated. For reference, and consistency, this section suggests non-  
806 normative choices for some of the various details involved in generating WSDL. For brevity, the following  
807 definitions apply:

- 808 • component name = the value of the @name attribute of the component element containing the  
809 binding.ws element
- 810 • service name = the value of the @name attribute of the service element containing the binding.ws  
811 element
- 812 • binding name = the value of @name attribute of the binding.ws element, or the default if no @name  
813 attribute is present
- 814 • SOAP version = either "SOAP11" or "SOAP12" as appropriate

815 With those definitions in place, here are the suggested choices:

- 816 • wsdl:definitions/@name = <component name> + "." + <service name>
- 817 • wsdl:definitions/@targetNamespace = <structural URI for the service>
- 818 • import each WSDL 1.1 portType, rather than putting them inline
- 819 • wsdl:binding/@name = <binding name> + <SOAP version> + "Binding"
- 820 • wsdl:service/@name = <service name>
- 821 • wsdl:port/@name = <binding name> + <SOAP version> + "Port"

822  
823

## E. SCA Web Services Callback Protocol Message Examples (Non-Normative)

824 The message examples in this section are for a configuration that consists of a reference R that is wired  
825 to a Service S. S has a bidirectional interface and the binding used in both directions, forward and  
826 callback, is binding.ws configured for SOAP. The forward interface and the callback interface both contain  
827 a single one-way operation.

828 The following message exchanges take place between R and S:

- 829 1. R invokes the forward operation and sets the callback address to **RC1**. Let's call the message that  
830 invokes the forward operation R1. S then calls the callback operation twice. Let's call the callback  
831 messages S1 and S2
- 832 2. R invokes the forward operation again with the same callback address **RC1**. Let's call the message  
833 that invokes the forward operation R2. S then calls the callback operation once. Let's call the callback  
834 message S3.
- 835 3. R invokes the forward operation yet another time, but this time uses a difference callback address:  
836 **RC2**. Let's call the message that invokes the forward operation R3. S then calls the callback  
837 operation twice. Let's call the callback messages S4 and S5.

838 The messages R1, R2, R3, S1, S2, S3, S4 and S4 are shown. The namespace prefix 'soap' can be  
839 bound to either the SOAP 1.1 or SOAP 1.2 namespace. The 'wsa' prefix is bound to the WS-Addressing  
840 1.0 namespace.

841  
842

### R1:

```
843 <soap:Envelope ...>  
844 <soap:Header>  
845   <wsa:From>  
846     <wsa:Address>http://example.com/callback</wsa:Address>  
847     <wsa:ReferenceProperties>  
848       <myNS:SomeID>1</myNS:SomeID>  
849     </wsa:ReferenceProperties>  
850   </wsa:From>  
851   <wsa:MessageID>urn:uuid:f81d4fae-7dec-11d0-a765-  
852 00a0c91e6bf6</wsa:messageID>  
853   ...  
854 </soap:Header>  
855 <soap:Body>  
856   ...  
857 </soap:Body>  
858 </soap:Envelope>
```

859

### S1, S2:

```
861 <soap:Envelope ...>  
862 <soap:Header>  
863   <wsa:To>http://example.com/callback</wsa:To>  
864   <myNS:SomeID>1</myNS:SomeID>  
865   <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-  
866 bindings/ws/callback">urn:uuid:f81d4fae-7dec-11d0-a765-  
867 00a0c91e6bf6</wsa:RelatesTo>  
868   ...  
869 </soap:Header>  
870 <soap:Body>  
871   ...  
872 </soap:Body>  
873 </soap:Envelope>
```



874

875 **R2:**

```
876 <soap:Envelope ...>
877 <soap:Header>
878 <wsa:From>
879 <wsa:Address>http://example.com/callback</wsa:Address>
880 <wsa:ReferenceProperties>
881 <myNS:SomeID>1</myNS:SomeID>
882 </wsa:ReferenceProperties>
883 </wsa:From>
884 <wsa:MessageID>urn:uuid:f81d4fae-8dec-11d0-a765-
885 00a0c91e6bf6</wsa:messageID>
886 ...
887 </soap:Header>
888 <soap:Body>
889 ...
890 </soap:Body>
891 </soap:Envelope>
```

892

893 **S3:**

```
894 <soap:Envelope ...>
895 <soap:Header>
896 <wsa:To>http://example.com/callback</wsa:To>
897 <myNS:SomeID>1</myNS:SomeID>
898 <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
899 bindings/ws/callback">
900 urn:uuid:f81d4fae-8dec-11d0-a765-00a0c91e6bf6
901 </wsa:RelatesTo>
902 ...
903 </soap:Header>
904 <soap:Body>
905 ...
906 </soap:Body>
907 </soap:Envelope>
```

908

909 **R3:**

```
910 <soap:Envelope ...>
911 <soap:Header>
912 <wsa:From>
913 <wsa:Address>http://example.com/callback-other</wsa:Address>
914 <wsa:ReferenceProperties>
915 <myNS:SomeID>2</myNS:SomeID>
916 </wsa:ReferenceProperties>
917 </wsa:From>
918 <wsa:MessageID>urn:uuid:f81d4fae-9dec-11d0-a765-
919 00a0c91e6bf6</wsa:messageID>
920 ...
921 </soap:Header>
922 <soap:Body>
923 ...
924 </soap:Body>
925 </soap:Envelope>
```

926

927 **S4, S5:**

```

928 <soap:Envelope ...>
929   <soap:Header>
930     <wsa:To>http://example.com/callback-other</wsa:To>
931     <myNS:SomeID>2</myNS:SomeID>
932     <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
933     bindings/ws/callback">urn:uuid:f81d4fae-9dec-11d0-a765-
934     00a0c91e6bf6</wsa:RelatesTo>
935     ...
936   </soap:Header>
937   <soap:Body>
938     ...
939   </soap:Body>
940 </soap:Envelope>

```

## 941 E.1 Message Examples Using WS-MakeConnection

942 In this case the reference R cannot host a listener and uses WS-MakeConnection to poll for callback  
943 requests. The interaction between the two consists of reference R sending a forward request R4. When  
944 using HTTP, the HTTP response to R4 contains an empty entity body. This is followed by a  
945 MakeConnection message from the reference to the service. This is a polling message from the reference  
946 and establishes a connection. If the callback request is ready when the connection is established, the  
947 service sends a callback request S6 to the reference in the entity body of the HTTP response.

948

949 **R4:**

```

950 <soap:Envelope ...>
951   <soap:Header>
952     <wsa:From>
953       <wsa:Address>http://docs.oasis-open.org/ws-
954       rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-446655440010</wsa:Address>
955     </wsa:From>
956     <wsa:MessageID>urn:uuid:f81d4fae-10dec-11d0-a765-
957     00a0c91e6bf6</wsa:messageID>
958     ...
959   </soap:Header>
960   <soap:Body>
961     ...
962   </soap:Body>
963 </soap:Envelope>

```

964

965 **MakeConnection polling message (from R to S):**

```

966 <soap:Envelope ...>
967   <soap:Header>
968     <wsa:Action>http://docs.oasis-open.org/ws-
969     rx/wsmc/200702/MakeConnection</wsa:Action>
970     ...
971   </soap:Header>
972   <soap:Body>
973     <wsmc:MakeConnection>
974       <wsmc:Address>http://docs.oasis-open.org/ws-
975       rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-
976       446655440010</wsmc:Address>
977     </wsmc:MakeConnection>
978   </soap:Body>
979 </soap:Envelope>

```

980

981 **S6:**

```
982 <soap:Envelope ...>
983   <soap:Header>
984     <wsa:To>http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id=650e8400-
985     f29b-11d4-a716-446655440010</wsa:To>
986     <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
987     bindings/ws/callback">urn:uuid:f81d4fae-10dec-11d0-a765-
988     00a0c91e6bf6</wsa:RelatesTo>
989     ...
990   </soap:Header>
991   <soap:Body>
992     ...
993   </soap:Body>
994 </soap:Envelope>
```

995

## F. Acknowledgements (Non-Normative)

996 The following individuals have participated in the creation of this specification and are gratefully  
997 acknowledged:

998 **Participants:**

Participant Name	Affiliation
Bryan Aupperle	IBM
Ron Barack	SAP AG
Michael Beisiegel	IBM
Henning Blohm	SAP AG
David Booz	IBM
Martin Chapman	Oracle Corporation
Jean-Sebastien Delfino	IBM
Laurent Domenech	TIBCO Software Inc.
Jacques Durand	Fujitsu Limited
Mike Edwards	IBM
Billy Feng	Primeton Technologies, Inc.
Nimish Hathalia	TIBCO Software Inc.
Simon Holdsworth	IBM
Eric Johnson	Software Inc.
Uday Joshi	Oracle Corporation
Khanderao Kand	Oracle Corporation
Anish Karmarkar	Oracle Corporation
Nickolaos Kavantzas	Oracle Corporation
Mark Little	Red Hat
Ashok Malhotra	Oracle Corporation
Jim Marino	Individual
Jeff Mischkinsky	Oracle Corporation
Dale Moberg	Axway Software
Simon Nash	Individual
Sanjay Patil	SAP AG
Plamen Pavlov	SAP AG
Peter Peshev	SAP AG
Piotr Przybylski	IBM
Luciano Resende	IBM
Tom Rutt	Fujitsu Limited
Vladimir Savchenko	SAP AG
Scott Vorthmann	TIBCO Software Inc.
Tim Watson	Oracle Corporation
Owen Williams	Avaya, Inc.
Prasad Yendluri	Software AG, Inc.

## G. Revision History (Non-Normative)

1000 [optional; should not be included in OASIS Standards]

Revision	Date	Editor	Changes Made
1	2007-09-25	Anish Karmarkar	Applied the OASIS template + related changes to the Submission
2	2008-04-02	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Partially applied the resolution of issue 14 in the conformance section.</li> <li>* Applied resolution to issue 9.</li> <li>* Applied resolution to issue 15.</li> <li>* Applied resolution to issue 16.</li> <li>* Applied resolution to issue 10.</li> <li>* Applied resolution to issue 8.</li> <li>* Applied resolution to issue 3.</li> </ul>
3	2008-06-12	Simon Holdsworth	<ul style="list-style-type: none"> <li>* Completed application of resolution to issue 10</li> <li>* Applied most of the editorial changes from Eric Johnson's review</li> </ul>
4	2008-08-13	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Applied rest of Eric Johnson's ed review comments.</li> <li>* Applied resolution of issue 13.</li> <li>* Reapplied resolution of issue 15 (it was not applied correctly before)</li> <li>* Applied resolution of issue 19.</li> <li>* Applied resolution of issue 30.</li> <li>* Applied resolution of issue 32.</li> <li>* Applied resolution of issue 36.</li> <li>* Applied resolution of issue 38.</li> </ul>
cd01-rev1	2008-10-16	Simon Holdsworth	Applied resolution of issue 41.
cd01-rev2	2008-10-20	Anish Karmarkar	Added rfc2119 statements.
cd01-rev3	2008-11-19	Anish Karmarkar	Incorporated feedback from Bryan, Eric & Dave
cd01-rev3	2008-12-02	Anish Karmarkar	Removed 'required' word associated with description of pseudo-schema + changed section 2.6 (wsdl extensibility) per the TC decision. Both of these were associated with issue 51 (2119 stmts)
cd01-rev5	2009-02-06	Simon Holdsworth	<ul style="list-style-type: none"> <li>Applied resolution of issue 11</li> <li>Applied resolution of issue 49</li> <li>Applied action item 20080904-1</li> </ul>
cd02	2009-02-16	Simon Holdsworth	Renamed, applied editorial issues

cd02-rev1	2009-06-02	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Applied resolution of issue 61 by using the document at <a href="http://www.oasis-open.org/apps/org/workgroup/sca-bindings/download.php/32160/sca-binding-ws-1.1-spec-cd02-issue61-rev3.doc">http://www.oasis-open.org/apps/org/workgroup/sca-bindings/download.php/32160/sca-binding-ws-1.1-spec-cd02-issue61-rev3.doc</a> as the base document.</li> <li>* Updated NS URI (Applied action item 20090311-2).</li> <li>* Updated Copyright statement in various places.</li> <li>* Updated schema per <a href="http://lists.oasis-open.org/archives/sca-bindings/200903/msg00057.html">http://lists.oasis-open.org/archives/sca-bindings/200903/msg00057.html</a> (Applied action item 20090312-1).</li> <li>* Applied resolution of issue 23, 25, 43, 54, 55, 64.</li> <li>* Replaced 3 occurrences of 'required' with 'specified'.</li> <li>* Recreated all bookmarks, cross-references, and conformance item table.</li> </ul>
cd02-rev2	2009-06-09	Anish Karmarkar	Ed. fixes. Changed the way the crossrefs/bookmarks for RFC2119 keywords work. Fixed a few references.
cd02-rev3	2009-06-11	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Removed ':' from 40005, reformatted 40006/40007.</li> <li>* minor ed changes pointed out by SimonN.</li> <li>* minor formatting changes.</li> <li>* modified BWS20018 to remove the first sentence.</li> </ul>
cd02-rev4	2009-06-17	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Not fixed in this rev, but issue 57 resolution was applied in previous rev.</li> <li>* Added list of participants in the Ack section.</li> <li>* Ed changes pointed out by Eric.</li> </ul>
cd02-rev5	2009-06-22	Anish Karmarkar	* Port of the fix made in JMS/JCA binding for issues 74/75. Specifically SCA WS Binding XML document requirements were made less vague (by referring to attributes/elements)
cd02-rev6	2009-06-24	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Applied resolution of issue 76, 79, 82.</li> <li>* Some very minor ed changes.</li> <li>* Reverted the document naming scheme to the old scheme.</li> </ul>
cd02-rev7	2009-07-01	Simon Holdsworth	<ul style="list-style-type: none"> <li>* Applied resolution of issue 2</li> <li>* Fixed application of resolution of issue 76</li> </ul>
cd03	2009-07-01	Simon Holdsworth	Renamed for cd03
cd03-rev1	2010-02-07	Bryan	Added table #, snippet #, etc.

cd03-rev2	2010-03-10	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Updated 'Notices' section for trademarks</li> <li>* Applied resolution of issue 99 points 9, 10, 16</li> <li>* Added references per <a href="http://lists.oasis-open.org/archives/sca-bindings/200912/msg00013.html">http://lists.oasis-open.org/archives/sca-bindings/200912/msg00013.html</a></li> <li>* Applied resolution of issue 84, 86, 91, 92, 116, 117, 118, 119</li> <li>* Updated NS URI from 200903 to 200912</li> </ul>
cd03-rev3	2010-03-31	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Updated schema appendix title to include "1.1"</li> <li>* Applied resolution of issue 124</li> <li>* Ed changes associated with issue 124 resolution</li> </ul>
cd03-rev4	2010-04-22	Anish Karmarkar	<ul style="list-style-type: none"> <li>* Fixed ed issues pointed out at <a href="http://lists.oasis-open.org/archives/sca-bindings/201004/msg00004.html">http://lists.oasis-open.org/archives/sca-bindings/201004/msg00004.html</a></li> </ul>

1001