



Service Component Architecture Web Service Binding Specification Version 1.1

Committee Draft 02 Revision **54** + issue 2 rev **65**

22nd17th June, 2009

Specification URIs:

This Version:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-binding-ws-1.1-spec-cd02.html>
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-binding-ws-1.1-spec-cd02.doc>
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-binding-ws-1.1-spec-cd02.pdf>
(Authoritative)

Previous Version:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd01.html>
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd01.doc>
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd01.pdf> (Authoritative)

Latest Version:

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

Latest Approved Version:

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:

- Service Component Architecture Assembly Model Specification Version 1.1
- Service Component Architecture Policy Framework Specification Version 1.1

Declared XML Namespace(s):

<http://docs.oasis-open.org/ns/opencsa/sca/200903>

Abstract:

The SCA Web Service binding specified in this document applies to the services and references of an SCA composite. 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 allows one to specify enough information to generate one. When an existing WSDL binding is not referenced, rules defined in this document allow one 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, 2009. 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.

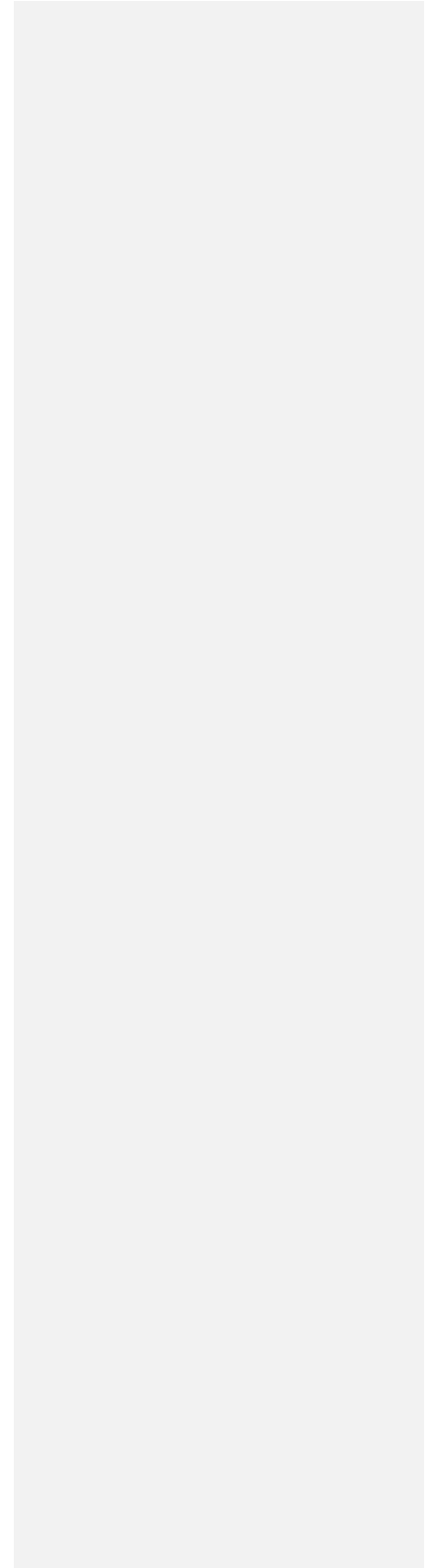
OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither 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" is a trademark 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.....	7
1.4	Naming Conventions.....	7
2	Web Service Binding Schema.....	9
2.1	Compatibility of SCA Service Interfaces and WSDL portTypes.....	11
2.2	Endpoint URI resolution.....	12
2.3	Interface mapping.....	12
2.4	Production of WSDL description for an SCA service.....	12
2.5	Additional binding configuration data.....	13
2.6	Web Service Binding and SOAP Intermediaries.....	13
2.7	Support for WSDL extensibility.....	13
2.8	Intents listed in the bindingType.....	13
2.9	Intents and binding configuration.....	14
3	Web Service Binding Examples.....	15
3.1	Example Using WSDL documents.....	15
3.2	Examples Without a WSDL Document.....	16
4	Transport Binding.....	18
4.1	Intents.....	18
4.2	Default Transport Binding Rules.....	18
4.2.1	WS-I Basic Profile Alignment.....	18
4.2.2	Default Transport Binding Rules.....	18
5	Implementing SCA Callbacks using Web Services.....	20
5.1	SCA Web Services Callback Protocol.....	20
5.2	SCA Web Services Callback with WS-MakeConnection Protocol.....	21
5.3	Policy Assertion for SCA Web Services Callback Protocol.....	21
5.3.1	Assertion Model.....	22+
5.3.2	Normative Outline.....	22
5.3.3	Assertion Attachment.....	22
5.3.4	Assertion Example.....	22
5.3.5	Security Considerations.....	23
6	Conformance.....	24
6.1	SCA WS Binding XML Document.....	24
6.2	SCA Runtime.....	24
A.	Web Services XML Binding Schema: sca-binding-webservice.xsd.....	25
B.	SCA Web Services Callback Protocol Policy Assertion XML Schema: sca-binding-webservice-callback.xsd.....	26
C.	Conformance Items.....	27
D.	Appendix - WSDL Generation.....	31
E.	SCA Web Services Callback Protocol Message Examples.....	32
E.1	Message Examples Using WS-MakeConnection.....	34
F.	Acknowledgements.....	36

G. Revision History.....38



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 allow one 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**]{**WSI-Profiles**}

Formatted: Font color: Auto

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.

Table 1-1 Prefixes and Namespaces used in this specification

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/wsdl/soap/"	Defined by WSDL 1.1 [WSDL11]
wsoap12	"http://schemas.xmlsoap.org/wsdl/soap12/"	Defined by [W11-SOAP12]
sca	"http://docs.oasis-open.org/ns/opencsa/sca/200903"	Defined by the SCA specifications

31

32 1.2 Normative References

- 33 **[RFC2119]** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
34 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 35 **[SCA-Assembly]** <http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec.pdf>
- 36 **[SCA-Policy]** <http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec.pdf>
- 37 **[SCA-JCAA]** <http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec.pdf>
- 38 **[WSDL11]** E. Christensen et al, *Web Service Description Language (WSDL) 1.1*,
39 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>, W3C Note, March 15 2001.
- 40 **[WSDL20]** Chinnici et al, *Web Service Description Language (WSDL) Version 2.0 Part 1: Core Language*, <http://www.w3.org/TR/2007/REC-wsdl20-20070626/>, W3C Recommendation, June 26 2007.
- 43 **[WSI-Profiles]** <http://www.ws-i.org/Profiles/BasicProfile-1.1.html>
44 <http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html>
45 <http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html>
46 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>
- 47 **[JAX-WS]** <http://jcp.org/en/jsr/detail?id=224>
- 48 **[SOAP11]** <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- 49 **[SOAP]** <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>
50 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- 51 **[SOAP12Adjuncts]** SOAP Version 1.2 Part 2: Adjuncts (Second Edition)
52 <http://www.w3.org/TR/soap12-part2/>
- 53 **[WS-Addr]** <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>
- 54 **[W11-SOAP12]** <http://www.w3.org/Submission/wsdl11soap12/>
- 55 **[WS-Addr-SOAP]** <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>
- 56 **[WS-MC]** <http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.html>
- 57 **[WS-Policy]** <http://www.w3.org/TR/2007/REC-ws-policy-20070904>
- 58 **[WS-PA]** <http://www.w3.org/TR/2007/REC-ws-policy-attach-20070904>
- 59

60 1.3 Non-Normative References

- 61 **[WSI-AP]** <http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html>
- 62 **[MTOM]** <http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/>
- 63 **[WS-Security]** <http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>
- 64

65 1.4 Naming Conventions

66 This specification follows some naming conventions for artifacts defined by the
67 specification. In addition to the conventions defined by section 1.3 of the Assembly
68 **[SCA-Assembly]** specification, this specification adds three additional conventions:

- 69 1. Where the names of elements and attributes consist partially or wholly of
70 acronyms, the letters of the acronyms use the same case. When the acronym

71 appears at the start of the name of an element or an attribute, or after a period,
72 it is in lower case. If it appears elsewhere in the name of an element or an
73 attribute, it is in upper case. For example, an attribute might be named "uri" or
74 "jndiURL".

75 2. Where the names of types consist partially or wholly of acronyms, the letters of
76 the acronyms are in all upper case. For example, an XML Schema type might be
77 named "JCABinding" or "MessageID".

78 3. Values, including local parts of QName values, follow the rules for names of
79 elements and attributes as stated above, with the exception that the letters of
80 acronyms are in all upper case. For example, a value might be "JMSDefault" or
81 "namespaceURI".

2 Web Service Binding Schema

The Web Service binding element is defined by the following pseudo-schema.

```
<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>
```

- **/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 below 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 the reference MUST contain at least one port. If the wsdl.service form of wsdlElement is used on an SCA reference binding, the set of available ports for the reference MUST contain at least one port. [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

129 an error if there are no available ports that it supports. [BWS20005] When an
130 invocation is made using an SCA reference binding with the *wsdl.service* form of
131 *wsdlElement*, the SCA runtime MUST use exactly one port from the set of
132 available ports for the reference (with port selection on a per-invocation basis
133 permitted). When an invocation is made using an SCA reference binding with the
134 *wsdl.service* form of *wsdlElement*, the SCA runtime MUST use exactly one port
135 from the set of available ports for the reference (with port selection on a per-
136 invocation basis permitted). [BWS20006]

137 • Port:

138 <WSDL-namespace-URI>#wsdl.port(<service-name>/<port-name>)

139 If the binding is for an SCA service, the portType associated with the specified
140 WSDL port MUST be compatible with the SCA service interface as defined in
141 section 2.1, and the port MUST satisfy all the policy constraints of the
142 binding. [BWS20007] The SCA runtime MUST expose an endpoint for the specified
143 WSDL port, or raise an error if it does not support the WSDL port. [BWS20008] If
144 the binding is for an SCA reference, the portType associated with the specified
145 WSDL port MUST be a compatible superset of the SCA reference interface as
146 defined in the SCA Assembly Model specification [SCA-Assembly], and the port
147 MUST satisfy all the policy constraints of the binding. If the binding is for an SCA
148 reference, the portType associated with the specified WSDL port MUST be a
149 compatible superset of the SCA reference interface as defined in the SCA
150 Assembly Model specification [SCA-Assembly], and the port MUST satisfy all the
151 policy constraints of the binding. [BWS20009] The SCA runtime MUST use the
152 specified WSDL port for invocations made using the SCA reference, or raise an
153 error if it does not support the WSDL port. [BWS20010]

154 • Binding:

155 <WSDL-namespace-URI>#wsdl.binding(<binding-name>)

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

162 If the binding is for an SCA reference, the portType associated with the specified
163 WSDL binding MUST be a compatible superset of the SCA reference interface as
164 defined in the SCA Assembly Model specification [SCA-Assembly], and the
165 WSDL binding MUST satisfy all the policy constraints of the binding. If the binding
166 is for an SCA reference, the portType associated with the specified WSDL binding
167 MUST be a compatible superset of the SCA reference interface as defined in the
168 SCA Assembly Model specification [SCA-Assembly], and the WSDL binding
169 MUST satisfy all the policy constraints of the binding. [BWS20013] The SCA
170 runtime MUST use the specified WSDL binding for invocations made using the
171 SCA reference, or raise an error if it does not support the WSDL binding.
172 [BWS20014]

173 When the *wsdl.binding* form of *wsdlElement* is used, the endpoint address URI
174 for an SCA reference MUST be specified by either the *@uri* attribute on the
175 binding or a WS-Addressing *EndpointReference* element, except where the SCA
176 Assembly Model specification [SCA-Assembly] states that the *@uri* attribute can
177 be omitted. When the *wsdl.binding* form of *wsdlElement* is used, the endpoint
178 address URI for an SCA reference MUST be specified by either the *@uri* attribute

Formatted: Highlight

Formatted: Highlight

Formatted: Highlight

179 | on the binding or a WS-Addressing *EndpointReference* element, except where the
180 | SCA Assembly Model specification [SCA-Assembly] states that the @uri
181 | attribute can be omitted. [BWS20015]

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

184 | The @wsdl:wsdlLocation attribute MAY be specified by the binding in the event that
185 | the <WSDL-namespace-URI> in the 'endpoint' attribute is not dereferencable, or
186 | when the intended WSDL document is to be found at a different location than the
187 | one pointed to by the <WSDL-namespace-URI>. The @wsdl:wsdlLocation attribute
188 | MAY be specified by the binding in the event that the <WSDL-namespace-URI> in
189 | the 'endpoint' attribute is not dereferencable, or when the intended WSDL document
190 | is to be found at a different location than the one pointed to by the <WSDL-
191 | namespace-URI>. [BWS20016]

192 | If the @wsdl:wsdlLocation attribute is used the @wsdlElement attribute MUST also
193 | be specified. If the @wsdl:wsdlLocation attribute is used the @wsdlElement attribute
194 | MUST also be specified. [BWS20017] The semantics of this attribute are specified in
195 | Section 7.1 of WSDL 2.0 [WSDL20]. The value of the @wsdl:wsdlLocation attribute
196 | MUST identify an existing WSDL 1.1 document. [BWS20018]

197 | • **/binding.ws/wireFormat** – as defined in the SCA Assembly Specification [SCA-
198 | Assembly]. This specification does not define any new wireFormat elements.

199 | • **/binding.ws/operationSelector** – as defined in the SCA Assembly Specification
200 | [SCA-Assembly]. This specification does not define any new operationSelector
201 | elements.

202 | • **/binding.ws/endpointReference** – when present this element provides the WS-
203 | Addressing [WS-Addr] EndpointReference that specifies the endpoint for the service
204 | or reference.

205 | • **/binding.ws/@{any}** - this is an extensibility mechanism to allow extensibility via
206 | attributes.

207 | • **/binding.ws/any** – this is an extensibility mechanism to allow extensibility via
208 | elements.

209 | A binding.ws element MUST NOT contain more than one of any of the following: the
210 | @uri attribute; the @wsdlElement attribute referring to a WSDL port or to a WSDL
211 | service; the endpointReference element. [BWS20019]

212 | The endpoint address URI for an SCA service or the callback element of an SCA
213 | reference is determined as specified in section 2.2. For the callback element of an SCA
214 | service, the binding MUST NOT specify an endpoint address URI or a WS-Addressing
215 | EndpointReference. For the callback element of an SCA service, the binding MUST NOT
216 | specify an endpoint address URI or a WS-Addressing EndpointReference. [BWS20020]

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

220 | The SCA runtime SHOULD support the element <endpointReference>. [BWS20022] If an
221 | SCA runtime does not support the element <endpointReference>, then it MUST reject
222 | an SCA WS Binding XML document (as defined in Section 5.1) that contains the
223 | element. If an SCA runtime does not support the element <endpointReference>, then it
224 | MUST reject an SCA WS Binding XML document (as defined in Section 6.1) that contains
225 | the element. [BWS20023]

226 The <binding.ws> element MUST conform to the XML schema defined in sca-binding-
227 webservice.xsd. [BWS20024]

228 2.1 Compatibility of SCA Service Interfaces and WSDL portTypes

229 A WSDL portType is compatible with an SCA service interface if and only if all of the
230 following conditions are satisfied:

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

249 2.2 Endpoint URI resolution

250 This specification does not mandate any particular way to determine the URI for a web
251 services binding on an SCA service. An absolute URI can be indicated by the @uri
252 attribute, by the URI in a wsa:Address element within an endpointReference element, or
253 by the URI indicated in a WSDL port via a @wsdlElement attribute. Implementations
254 can use the specified URI as the service endpoint URI or they can use a different URI
255 which might include portions of the specified URI. For example, the service endpoint
256 URI might be produced by modifying any or all of the host name, the port number, and
257 a portion of the path.

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

263 The target address for a reference binding is defined as one of the following:

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

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

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

275 2.3 Interface mapping

276 When *binding.ws* is used on a service or reference with an interface that is not defined
277 by *interface.wsdl*, the SCA runtime MUST derive a WSDL portType for the service or
278 reference from the interface using the rules defined for that SCA interface type. When
279 *binding.ws* is used on a service or reference with an interface that is not defined by
280 *interface.wsdl*, the SCA runtime MUST derive a WSDL portType for the service or
281 reference from the interface using the rules defined for that SCA interface type.
282 [BWS20027]

283 An SCA runtime MUST raise an error if the interface on a service or reference element
284 with a *binding.ws* element does not map to a WSDL portType. An SCA runtime MUST
285 raise an error if the interface on a service or reference element with a *binding.ws*
286 element does not map to a WSDL portType. [BWS20028]

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

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

292

293 2.4 Production of WSDL description for an SCA service

294 Any service hosted by an SCA runtime with one or more web service bindings with HTTP
295 endpoints SHOULD return a WSDL description of the service in response to an HTTP GET
296 request with the "?wsdl" suffix to that HTTP endpoint. Any service hosted by an SCA
297 runtime with one or more web service bindings with HTTP endpoints SHOULD return a
298 WSDL description of the service in response to an HTTP GET request with the "?wsdl"
299 suffix to that HTTP endpoint. [BWS20029]

300 If none of the web service bindings for an SCA service have HTTP endpoints, then the
301 SCA runtime SHOULD provide some other means of obtaining the WSDL description of
302 the service. If none of the web service bindings for an SCA service have HTTP endpoints,
303 then the SCA runtime SHOULD provide some other means of obtaining the WSDL
304 description of the service. [BWS20030] This can include out of band mechanisms, for
305 example publication to a UDDI registry.

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

308

309 2.5 Additional binding configuration data

310 SCA runtime implementations MAY provide additional metadata that is associated with a
311 web service binding. SCA runtime implementations MAY provide additional metadata that
312 is associated with a web service binding. [BWS20031]

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

316

317 2.6 Web Service Binding and SOAP Intermediaries

318 The Web Service binding does not provide any direct or explicit support for SOAP
319 intermediaries **[SOAP]**.

320

321 2.7 Support for WSDL extensibility

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

328 **An SCA runtime MUST support the WSDL extensions defined in the namespace**
329 **associated with the prefix "sca" (as defined in section 1.1). An SCA runtime MUST**
330 **support the WSDL extensions defined in the namespace associated with the prefix "sca"**
331 **(as defined in section 1.1); [BWS20032]**

332 **The SCA runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP**
333 **[WSDL11], as identified by the WSDL element wsoap11:binding that has the**
334 **@transport attribute with a value of "http://schemas.xmlsoap.org/soap/http". The SCA**
335 **runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP**
336 **[WSDL11], as identified by the WSDL element wsoap11:binding that has the**
337 **@transport attribute with a value of "http://schemas.xmlsoap.org/soap/http";**
338 **[BWS20033]**

Formatted: Highlight

339 **The SCA runtime SHOULD support the WSDL 1.1 binding extension for SOAP 1.2 over**
340 **HTTP [W11-SOAP12], as identified by the WSDL element wsoap12:binding that has**
341 **the @transport attribute with a value of "http://schemas.xmlsoap.org/soap/http". The**
342 **SCA runtime SHOULD support the WSDL 1.1 binding extension for SOAP 1.2 over HTTP**
343 **[W11-SOAP12], as identified by the WSDL element wsoap12:binding that has the**
344 **@transport attribute with a value of "http://schemas.xmlsoap.org/soap/http";**
345 **[BWS20034]**

Formatted: Highlight

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

350 2.8 Intents listed in the bindingType

351 This specification places no requirements on the intents that are listed as either
352 @alwaysProvides or @mayProvides in the bindingType for binding.ws.

353 2.9 Intents and binding configuration

354 This binding mandates support for SOAP 1.1 and encourages SOAP 1.2 support. **The**
355 **<bindingType> element associated with this binding MUST include the SOAP_1_1 intent**
356 **in its @mayProvides or @alwaysProvides attributes. [BWS20035] The <bindingType>**

357 element associated with this binding SHOULD include the SOAP.1_2 intent in its
358 @mayProvides attribute. [BWS20036] For more details on the <bindingType> element
359 see [SCA-Policy].

360 | The SCA runtime MUST raise an error if a web service binding is configured with a policy
361 | intent(s) that conflicts with the binding instance's configuration. The SCA runtime MUST
362 | raise an error if a web service binding is configured with a policy intent(s) that conflicts
363 | with the binding instance's configuration. [BWS20037]

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

366 3 Web Service Binding Examples

367 The following snippets show the sca.composite file for the MyValueComposite file
368 containing the service element for the MyValueService and reference element for the
369 StockQuoteService. Both the service and the reference use a Web Service binding.
370

371 3.1 Example Using WSDL documents

372 This example shows a service and reference using the SCA Web Service binding, using
373 existing WSDL documents in both cases. In each case there is a single binding element,
374 whose name defaults to the service/reference name.

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

377 The first reference's binding is defined by the specified WSDL service in the WSDL
378 document at the given location. The reference can use any of the WSDL service's ports
379 to invoke the target service. The second reference's binding is defined by the specified
380 WSDL binding. The specific endpoint URI to be invoked is provided via the *@uri*
381 attribute.
382

```
383 <?xml version="1.0" encoding="ASCII"?>  
384 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200903"  
385           name="MyValueComposite">  
386   <service name="MyValueService">  
387     <interface.java interface="services.myvalue.MyValueService"/>  
388     <binding.ws wsdlElement="http://www.example.org/MyValueService#  
389  
390 wsdl.binding(MyValueService/MyValueServiceSOAP)"/>  
391     ...  
392   </service>  
393  
394   ...  
395  
396   <reference name="StockQuoteReference1">  
397     <interface.java interface="services.stockquote.StockQuoteService"/>  
398     <binding.ws wsdlElement="http://www.example.org/StockQuoteService#  
399                 wsdl.service(StockQuoteService) "  
400     wsdl:wsdlLocation="http://www.example.org/StockQuoteService  
401                       http://www.example.org/StockQuoteService.wsdl"/>  
402   </reference>  
403  
404   <reference name="StockQuoteReference2">  
405     <interface.java interface="services.stockquote.StockQuoteService"/>  
406     <binding.ws wsdlElement="http://www.example.org/StockQuoteService#  
407                 wsdl.binding(StockQuoteBinding) "  
408     wsdl:wsdlLocation="http://www.example.org/StockQuoteService  
409                       http://www.example.org/StockQuoteService.wsdl"  
410     uri="http://www.example.org/StockQuoteService5"/>  
411   </reference>  
412 </composite>
```


413 3.2 Examples Without a WSDL Document

414 The next example shows the simplest form of the binding element without WSDL
415 document, assuming all defaults for portType mapping and SOAP binding synthesis. The
416 service and reference each have a single binding element, whose name defaults to the
417 service/reference name.

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

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

```
426  
427 <?xml version="1.0" encoding="ASCII"?>  
428 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200903"  
429     name="MyValueComposite">  
430  
431     <service name="MyValueService">  
432         <interface.java interface="services.myvalue.MyValueService"/>  
433         <binding.ws/>  
434         ...  
435     </service>  
436  
437     ...  
438  
439     <reference name="StockQuoteService">  
440         <interface.java interface="services.stockquote.StockQuoteService"/>  
441         <binding.ws uri="http://www.example.org/StockQuoteService"/>  
442     </reference>  
443 </composite>
```

444 The next example shows the use of the binding element without a WSDL document, with
445 multiple SOAP bindings with non-default values. The SOAP 1.2 binding name defaults to
446 the service name, the SOAP 1.1 binding is given an explicit name. The reference has a
447 web service binding which uses SOAP 1.2, but otherwise uses all the defaults for SOAP
448 binding. The reference binding name defaults to the reference name.

```
450  
451 <?xml version="1.0" encoding="ASCII"?>  
452 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200903"  
453     name="MyValueComposite">  
454  
455     <service name="MyValueService">  
456         <interface.java interface="services.myvalue.MyValueService"/>  
457         <binding.ws name="MyValueServiceSOAP11" requires="SOAP.1_1"/>  
458         <binding.ws requires="SOAP.1_2"/>  
459         ...  
460     </service>  
461  
462     ...  
463  
464     <reference name="StockQuoteService">  
465         <interface.java interface="services.stockquote.StockQuoteService"/>  
466         <binding.ws uri="http://www.example.org/StockQuoteService"  
467             requires="SOAP.1_2"/>  
468     </reference>
```

469
470

```
</composite>
```

471 4 Transport Binding

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

477 4.1 Intents

478 So as to narrow the range of choices for how messages are carried, the following policy
479 intents affect the transport binding:

- 480 • SOAP
481 **When the SOAP intent is required, the SCA runtime MUST transmit and receive**
482 **messages using SOAP. One or more SOAP versions can be used. When the SOAP**
483 **intent is required, the SCA runtime MUST transmit and receive messages using**
484 **SOAP. One or more SOAP versions can be used. [BWS40001]**
- 485 • SOAP.1_1
486 **When the SOAP.1_1 intent is required, the SCA runtime MUST transmit and receive**
487 **messages using only SOAP 1.1. When the SOAP.1_1 intent is required, the SCA**
488 **runtime MUST transmit and receive messages using only SOAP 1.1. [BWS40002]**
- 489 • SOAP.1_2
490 **When the SOAP.1_2 intent is required, the SCA runtime MUST transmit and receive**
491 **messages using only SOAP 1.2. When the SOAP.1_2 intent is required, the SCA**
492 **runtime MUST transmit and receive messages using only SOAP 1.2. [BWS40003]**

493 4.2 Default Transport Binding Rules

494 4.2.1 WS-I Basic Profile Alignment

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

- 499 • that every message part references an XML Schema type (rpc-literal pattern)
- 500 • or that every message references exactly zero or one XML Schema elements
501 (document-literal pattern)

502 **For an SCA service or reference element, the portType from the service's or reference's**
503 **interface or derived from that interface MUST follow either the rpc-literal pattern or the**
504 **document-literal pattern. For an SCA service or reference element, the portType from the**
505 **service's or reference's interface or derived from that interface MUST follow either the**
506 **rpc-literal pattern or the document-literal pattern. [BWS40004]**

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

510 4.2.2 Default Transport Binding Rules

511 The following defines the **default transport binding rules** for the Web Service binding:

- 512
- HTTP-based transfer protocol;
- 513
- SOAP 1.1 binding;
- 514
- "literal" format as described in section 3.5 of **[WSDL11]**;
- 515
- Either the document literal or rpc literal pattern, depending on the service or
- 516
- reference interface as described in section 4.2.1;
- 517
- For document literal pattern, each message uses "document" style, as per
- 518
- section 3.5 of **[WSDL11]**;
- 519
- For rpc-literal pattern, each message uses "rpc" style, as per section 3.5
- 520
- of **[WSDL11]** and the child elements of the SOAP Body element are
- 521
- namespace qualified with a non-empty namespace name;
- 522
- For SOAP 1.1 messages, the SOAPAction HTTP header described in section 6.1.1
- 523
- of **[SOAP11]** represents the empty string, in quotes ("");
- 524
- For SOAP 1.2 messages, the SOAP Action feature described in section 6.5 of
- 525
- [SOAP12Adjuncts]** does not appear;
- 526
- All WSDL message parts are carried in the SOAP body.

527 ~~In the event that the transport details are not otherwise determined, an SCA runtime~~
528 ~~MUST enable the default transport binding rules. In the event that the transport details~~
529 ~~are not otherwise determined, an SCA runtime MUST enable the default transport~~
530 ~~binding rules. [BWS40005]~~

531 ~~When using the default transport binding rules, the SCA runtime MAY provide additional~~
532 ~~WSDL bindings, unless policy is applied that explicitly restricts this. When using the~~
533 ~~default transport binding rules, the SCA runtime MAY provide additional WSDL bindings,~~
534 ~~unless policy is applied that explicitly restricts this. [BWS40006]~~

535 ~~When using the default transport binding rules with the rpc-literal pattern, the SCA~~
536 ~~runtime SHOULD use the structural URI associated with the binding as the namespace of~~
537 ~~the child elements of the SOAP body element. When using the default transport binding~~
538 ~~rules with the rpc-literal pattern, the SCA runtime SHOULD use the structural URI~~
539 ~~associated with the binding as the namespace of the child elements of the SOAP body~~
540 ~~element. [BWS40007]~~

541 5 Implementing SCA Callbacks using Web Services

542 5.1 SCA Web Services Callback Protocol

543 This section defines the SCA Web Services callback protocol that can be used to
544 implement a bidirectional interface in conjunction with the Web Services binding. For
545 examples of wire messages exchanged when using this protocol see ~~the~~ Appendix EE.

546 To implement the SCA Web Services Callback Protocol, an SCA binding follows the
547 following rules.

- 548 1. Every request message that invokes the forward interface MUST contain a
549 Callback EPR. [BWS50002] If the request message contains the `wsa:From` SOAP
550 header block then the `wsa:From` header block specifies the Callback EPR. If the
551 `wsa:From` header block is not present then the `wsa:ReplyTo` header block
552 specifies the Callback EPR.

553 If the Callback EPR's [address] value is
554 "`http://www.w3.org/2005/08/addressing/anonymous`" or
555 "`http://www.w3.org/2005/08/addressing/none`" then the SCA runtime MUST
556 generate the Invalid Addressing Header fault as specified in Section 6.4.1 of
557 [WS-Addr-SOAP]. If the Callback EPR's [address] value is
558 "`http://www.w3.org/2005/08/addressing/anonymous`" or
559 "`http://www.w3.org/2005/08/addressing/none`" then the SCA runtime MUST
560 generate the Invalid Addressing Header fault as specified in Section 6.4.1 of
561 [WS-Addr-SOAP]. [BWS50004] Such a fault can include additional
562 [Subsubcode] `wsa:OnlyNonAnonymousAddressSupported`.

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

- 567 3. When the service implementation invokes the callback interface, it MUST use the
568 Callback EPR from a request message that invoked the forward interface. When
569 the service implementation invokes the callback interface, it MUST use the
570 Callback EPR from a request message that invoked the forward interface, as
571 specified in BWS50003. [BWS50005] Once the Callback EPR is selected, the SCA
572 runtime MUST follow the rules defined in Section 3.3 of [WS-Addr] to invoke
573 operations on the callback interface. Once the Callback EPR is selected, the SCA
574 runtime MUST follow the rules defined in Section 3.3 of [WS-Addr] to invoke
575 operations on the callback interface. [BWS50006]

576 When the service invokes the callback interface, if the request message from which the
577 Callback EPR was obtained contained the `wsa:MessageID` SOAP header block, the SCA
578 runtime MUST include a `wsa:RelatesTo` SOAP header block in the callback
579 message. When the service invokes the callback interface, if the request message from
580 which the Callback EPR was obtained contained the `wsa:MessageID` SOAP header block,
581 the SCA runtime MUST include a `wsa:RelatesTo` SOAP header block in the callback
582 message. [BWS50007] The `wsa:RelatesTo` SOAP header block MUST have the
583 relationship type value of "`http://docs.oasis-open.org/opencsa/sca-`
584 `bindings/ws/callback`" and the related message id MUST be the `wsa:MessageID` of the
585 message from which the Callback EPR was obtained. The `wsa:RelatesTo` SOAP header

Formatted: Highlight

Formatted: Highlight

586 | block MUST have the relationship type value of "[http://docs.oasis-](http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback)
587 | [open.org/opencsa/sca-bindings/ws/callback](http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback)" and the related message id MUST be
588 | the `wsa:MessageID` of the message from which the Callback EPR was obtained.
589 | [BWS50008]

590 | If the request message from which the Callback EPR was obtained did not contain the
591 | `wsa:MessageID` SOAP header block, the SCA runtime MUST NOT include a
592 | `wsa:RelatesTo` SOAP header block with a relationship type value of
593 | "<http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback>" in the callback
594 | message. If the request message from which the Callback EPR was obtained did not
595 | contain the `wsa:MessageID` SOAP header block, the SCA runtime MUST NOT include a
596 | `wsa:RelatesTo` SOAP header block with a relationship type value of
597 | "<http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback>" in the callback
598 | message. [BWS50009]

599 | When a service that offers a bidirectional interface is invoked, depending on the
600 | semantics and/or implementation of the service, it is possible that the service might
601 | invoke the callback interface before the forward operation ends. In such cases, it is
602 | necessary for the binding on the reference-side to be listening for callback request(s)
603 | from the service, before the forward operation request is sent on the wire to the service,
604 | and continue listening as long as callback requests are expected. It is possible that
605 | before the response to the forward request is sent a response to one or more callback
606 | requests are required by the service.

607 | **5.2 SCA Web Services Callback Protocol with WS-MakeConnection** 608 | **Protocol**

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

617 | When an SCA runtime implements the SCA Web Services Callback protocol in
618 | conjunction with WS-MakeConnection, it has to adhere to the rules described for the
619 | SCA Web Services Callback Protocol and also to those of WS-MakeConnection.

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

624 | The unique-String value is a globally unique value such as a UUID, as defined by the
625 | WS-MakeConnection specification.

626 | When the service implementation invokes the callback interface, it uses the Callback EPR
627 | from a request message that invoked the forward interface, and the callback request
628 | message is sent as the response to a `wsmc:MakeConnection` message that contains the
629 | `wsmc:Address` value that matches the MakeConnection Anonymous URI in the Callback
630 | EPR.

631 | When a service that offers a bidirectional interface is invoked using WS-MakeConnection
632 | Anonymous URI as the value for the Callback EPR address, depending on the semantics

633 and/or implementation of the service, it is possible that the service might invoke the
634 callback interface before the forward operation ends. In such cases, it is necessary for
635 the binding on the reference-side to start polling for callback request(s) from the
636 service, before or right after the forward operation request is sent and before a response
637 is received, and continue polling as long as callback requests are expected. It is possible
638 that before the response to the forward request is sent a response to one or more
639 callback requests are required by the service.

640 5.3 Policy Assertion for SCA Web Services Callback Protocol

641 WS-Policy Framework **[WS-Policy]** and WS-Policy Attachment **[WS-PA]** collectively
642 define a framework, model and grammar for expressing the requirements, and general
643 characteristics of entities in an XML Web services-based system. To enable a Web
644 service client and a Web service to describe their requirements for implementing SCA
645 Web Services Callback Protocol (~~see SCA Web Services Callback Protocol~~), this
646 specification defines a single policy assertion that leverages the WS-Policy framework.

647 5.3.1 Assertion Model

648 The WSCallback policy assertion indicates that the Web service client and the Web
649 service **MUST use SCA Web Services Callback Protocol to implement callbacks.**
650 **[BWS50010]** Specifically, the protocol determines the requirements on the forward
651 request message, the EPR used for callbacks and the requirements on the callback
652 request message.

653 5.3.2 Normative Outline

654 The normative outline for the WSCallback assertion is:

```
655 <sca:WSCallback ...>  
656 ...  
657 </sca:WSCallback>
```

658
659 The following describes the content model of the WSCallback element.

- 660 • `/sca:WSCallback`: A policy assertion that specifies that [SCA Web Services](#)
661 `WSCallback` protocol is used when sending messages.

662 5.3.3 Assertion Attachment

663 The WSCallback policy assertion is allowed to have the following Policy Subjects **[WS-**
664 **PA]**:

- 665 • Endpoint Policy Subject

666 WS-PolicyAttachment defines a set of WSDL/1.1 policy attachment points for each of the
667 above Policy Subjects. ~~Since a WSCallback policy assertion specifies a concrete behavior,~~
668 ~~it cannot be attached to the abstract WSDL policy attachment points. Since a~~
669 ~~WSCallback policy assertion specifies a concrete behavior, it MUST NOT be attached to~~
670 ~~the abstract WSDL policy attachment points. [BWS50012]~~

671 ~~The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects~~
672 ~~allowed for a WSCallback policy assertion but which MUST NOT have WSCallback policy~~
673 ~~assertions attached: wsdl:portType~~The following is the list of WSDL/1.1 elements whose
674 scope contains the Policy Subjects allowed for a WSCallback policy assertion but which
675 ~~MUST NOT have WSCallback policy assertions attached: wsdl:portType [BWS50013]~~

Formatted: Not Highlight

Formatted: Not Highlight

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

- 679 • wsdl:port
- 680 • wsdl:binding

681 5.3.4 Assertion Example

682 The example below shows the use of the WSCallback policy assertion in a WSDL
683 document.

684

```
685 (01)<wsdl:definitions  
686 (02)   targetNamespace="example.com"  
687 (03)   xmlns:tns="example.com"  
688 (04)   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"  
689 (05)   xmlns:wsp="http://www.w3.org/ns/ws-policy"  
690 (06)   xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200903"  
691 (07)   xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
692 wssecurity-utility-1.0.xsd">  
693 (08)  
694 (09)   <wsp:UsingPolicy wsdl:required="true" />  
695 (10)  
696 (11)   <wsp:Policy wsu:Id="MyPolicy" >  
697 (12)     <sca:WSCallback/>  
698 (13)   </wsp:Policy>  
699 (14)  
700 (15)   <!-- omitted elements -->  
701 (16)  
702 (17)   <wsdl:binding name="MyBinding" type="tns:MyPortType" >  
703 (18)     <wsp:PolicyReference URI="#MyPolicy" />  
704 (19)   <!-- omitted elements -->  
705 (20)   </wsdl:binding>  
706 (21)  
707 (22)</wsdl:definitions>
```

708

709 Line (09) in the example above indicates that WS-Policy is in use as a required
710 extension. Lines (11-13) are a policy expression that includes a WSCallback policy
711 assertion (line 12) to indicate that SCA Web Services Callback protocol is used. Lines
712 (17-20) are a WSDL binding. Line (18) indicates that the policy in lines (11-13) applies
713 to this binding, specifically indicating that SCA Web Services Callback protocol is used
714 over all the messages in the binding.

715 5.3.5 Security Considerations

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

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

724

725 6 Conformance

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

729 There are two categories of artifacts for which this specification defines conformance:

- 730 a) SCA WS Binding XML Document
- 731 b) SCA Runtime

732 6.1 SCA WS Binding XML Document

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

736 An SCA WS Binding XML document **MUST** be a conformant SCA Composite Document or
737 a SCA ComponentType Document, as defined by the SCA Assembly specification **[SCA-**
738 **Assembly]**, and **MUST** comply with all [statements in Appendix C: Conformance Items](#)
739 [related to elements and attributes in an SCA WS Binding XML document, notably all](#)
740 ["MUST" statements have to be implemented the applicable requirements specified in this](#)
741 [specification](#).

742 6.2 SCA Runtime

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

- 745 1. The implementation **MUST** comply with all statements in Appendix B:
746 Conformance Items related to an SCA Runtime, except for those that originate
747 from Section 5, notably all "MUST" statements have to be implemented.
- 748 2. The implementation **MAY** support the SCA Web Services Callback Protocol. If it
749 does, it **MUST** comply with all statements in Appendix B: Conformance Items for
750 the SCA Web Services Callback Protocol.
- 751 3. The implementation **MAY** support the SCA Web Services Callback [Protocol](#) in
752 conjunction with the WS-MakeConnection-~~Protocol~~. If it does, it **MUST** comply
753 with all statements in Appendix B: Conformance Items for the SCA Web Services
754 Callback [Protocol](#) and it **MUST** comply with the requirements of WS-
755 MakeConnection-~~Protocol~~.
- 756 4. The implementation **MUST** conform to the SCA Assembly Model Specification
757 Version 1.1 **[SCA-Assembly]**, and to the SCA Policy Framework Version 1.1
758 **[SCA-Policy]**.
- 759 5. The implementation **MUST** reject a SCA WS Binding XML Document that is not
760 conformant per Section 6.1.

761
762

A. Web Services XML Binding Schema: sca-binding-webservice.xsd

763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (C) OASIS 2005, 2009. 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/200903"
  xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200903"
  xmlns:wsdl="http://www.w3.org/ns/wsdl-instance"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  elementFormDefault="qualified">

  <import namespace="http://www.w3.org/ns/wsdl-instance"
    schemaLocation="http://www.w3.org/2007/05/wsdl/wsdl20-
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-cd03.xsd"/>

  <element name="binding.ws" type="sca:WebServiceBinding"
    substitutionGroup="sca:binding"/>
  <complexType name="WebServiceBinding">
    <complexContent>
      <extension base="sca:Binding">
        <sequence>
          <element ref="sca:wireFormat"
            minOccurs="0" maxOccurs="1" />
          <element ref="sca:operationSelector"
            minOccurs="0" maxOccurs="1" />
          <element name="endpointReference"
            type="wsa:EndpointReference"
            minOccurs="0" maxOccurs="unbounded"/>
          <any namespace="##other" processContents="lax"
            minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="wsdlElement" type="anyURI" use="optional"/>
        <attribute ref="wsdl:wsdlLocation" use="optional"/>
        <anyAttribute namespace="##any" processContents="lax"/>
      </extension>
    </complexContent>
  </complexType>
</schema>
```

808
809
810

B. SCA Web Services Callback Protocol Policy Assertion XML Schema: sca-binding-webservice- callback.xsd

811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- (c) Copyright OASIS 2005, 2009. 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/200903"
  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>
```

C. Conformance Items

833 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]	If the wsdl.service form of wsdlElement is used on an SCA reference binding, the set of available ports for the reference MUST contain at least one port.
[BWS20005][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]	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][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][BWS20010]	The SCA runtime MUST use the specified WSDL port for invocations made using the SCA reference, 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, 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.
[BWS20016]	The <i>@wsdl:wsdlLocation</i> attribute MAY be specified by the binding in the event that the <i><WSDL-namespace-URI></i> in the 'endpoint' attribute is not dereferencable, or when the intended WSDL document is to be found at a different location than the one pointed to by the <i><WSDL-namespace-URI></i> .
[BWS20017]	If the <i>@wsdl:wsdlLocation</i> attribute is used the <i>@wSDL.Element</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>@wSDL.Element</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><binding.ws></i> element, namely <i>@name</i> , <i>@uri</i> , <i>@requires</i> , <i>@policySets</i> , <i>@wSDL.Element</i> , and <i>@wsdl:wsdlLocation</i> .
[BWS20022]	The SCA runtime SHOULD support the element <i><endpointReference></i> .
[BWS20023]	If an SCA runtime does not support the element <i><endpointReference></i> , then it MUST reject an SCA WS Binding XML document (as defined in Section 5.1) that contains the element.
[BWS20024]	The <i><binding.ws></i> element MUST conform to the XML schema defined in <i>sca-binding-webservice.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 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 to that HTTP endpoint.
[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

Formatted: Highlight

	WSDL description of the service.
[BWS20031]	SCA runtime implementations MAY provide additional metadata that is associated with a web service binding.
[BWS20032]	An SCA runtime MUST support the WSDL extensions defined in the namespace associated with the prefix "sca" (as defined in section 1.1).
[BWS20033]	The SCA runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP [WSDL11] [WSDL11] , 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 [W11-SOAP12] [W11-SOAP12] , 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.1_1 intent in its @mayProvides or @alwaysProvides attributes.
[BWS20036]	The <bindingType> element associated with this binding SHOULD include the SOAP.1_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.
[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] [BWS40002]	When the SOAP.1_1 intent is required, the SCA runtime MUST transmit and receive messages using only SOAP 1.1.
[BWS40003] [BWS40003]	When the SOAP.1_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 otherwise determined, an SCA runtime MUST enable the default transport binding rules.
[BWS40006]	When using the default transport binding rules, the SCA runtime MAY provide additional WSDL bindings, unless policy is applied that explicitly restricts this.
[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 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 SCA runtime MUST generate the Invalid Addressing Header fault as specified in Section 6.4.1 of [WS-Addr-SOAP] [WS-Addr-SOAP] .
[BWS50005]	When the service implementation invokes the callback interface, it MUST use the Callback EPR from a request message that invoked the forward interface.

Formatted: Highlight

Formatted: Highlight

Formatted: Highlight

	as specified in BWS50003.
[BWS50006]	Once the Callback EPR is selected, the SCA runtime MUST follow the rules defined in Section 3.3 of [WS-Addr][WS-Addr] to invoke operations on the callback interface.
[BWS50007][BWS50007]	When the service invokes the callback interface, if the request message from which the Callback EPR was obtained contained the wsa:MessageID SOAP header block, the SCA runtime MUST include a wsa:RelatesTo SOAP header block in the callback message.
[BWS50008]	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.
[BWS50009][BWS50009]	If the request message from which the Callback EPR was obtained did not contain the wsa:MessageID SOAP header block, the SCA runtime MUST NOT include a wsa:RelatesTo SOAP header block with a relationship type value of "http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback" in the callback message.
[BWS50010][BWS50010]	The WSCallback policy assertion indicates that the Web service client and the Web service MUST use SCA Web Services Callback Protocol to implement callbacks.
[BWS50012]	Since a WSCallback policy assertion specifies a concrete behavior, it MUST NOT be attached to the abstract WSDL policy attachment points.
[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: wsd:portType
[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.

Formatted: Highlight

834 **D. Appendix - WSDL Generation**

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

- 840 • component name = the value of the @name attribute of the component element
841 containing the binding.ws element
- 842 • service name = the value of the @name attribute of the service element
843 containing the binding.ws element
- 844 • binding name = the value of @name attribute of the binding.ws element, or the
845 default if no @name attribute is present
- 846 • SOAP version = either "SOAP11" or "SOAP12" as appropriate

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

- 848 • wsdl:definitions/@name = <component name> + "." + <service name>
- 849 • wsdl:definitions/@targetNamespace = <structural URI for the service>
- 850 • import each WSDL 1.1 portType, rather than putting them inline
- 851 • wsdl:binding/@name = <binding name> + <SOAP version> + "Binding"
- 852 • wsdl:service/@name = <service name>
- 853 • wsdl:port/@name = <binding name> + <SOAP version> + "Port"

854
855

E. SCA Web Services Callback Protocol Message Examples

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

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

- 861 1. R invokes the forward operation and sets the callback address to **RC1**. Let's call
862 the message that invokes the forward operation R1. S then calls the callback
863 operation twice. Let's call the callback messages S1 and S2
- 864 2. R invokes the forward operation again with the same callback address **RC1**. Let's
865 call the message that invokes the forward operation R2. S then calls the callback
866 operation once. Let's call the callback message S3.
- 867 3. R invokes the forward operation yet another time, but this time uses a difference
868 callback address: **RC2**. Let's call the message that invokes the forward operation
869 R3. S then calls the callback operation twice. Let's call the callback messages S4
870 and S5.

871 The messages R1, R2, R3, S1, S2, S3, S4 and S5 are listed below. The namespace
872 prefix 'soap' can be bound to either the SOAP 1.1 or SOAP 1.2 namespace. The 'wsa'
873 prefix is bound to the WS-Addressing 1.0 namespace.

874 **R1:**

```
875 <soap:Envelope ...>  
876 <soap:Header>  
877 <wsa:From>  
878 <wsa:Address>http://example.com/callback</wsa:Address>  
879 <wsa:ReferenceProperties>  
880 <myNS:SomeID>1</myNS:SomeID>  
881 </wsa:ReferenceProperties>  
882 </wsa:From>  
883 <wsa:MessageID>urn:uuid:f81d4fae-7dec-11d0-a765-  
884 00a0c91e6bf6</wsa:messageID>  
885 ...  
886 </soap:Header>  
887 <soap:Body>  
888 ...  
889 </soap:Body>  
890 </soap:Envelope>
```

892

893 **S1, S2:**

```
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">urn:uuid:f81d4fae-7dec-11d0-a765-
900   00a0c91e6bf6</wsa:RelatesTo>
901   ...
902 </soap:Header>
903 <soap:Body>
904   ...
905 </soap:Body>
906 </soap:Envelope>
907
```

908

909

R2:

```
910 <soap:Envelope ...>
911 <soap:Header>
912   <wsa:From>
913     <wsa:Address>http://example.com/callback</wsa:Address>
914     <wsa:ReferenceProperties>
915       <myNS:SomeID>1</myNS:SomeID>
916     </wsa:ReferenceProperties>
917   </wsa:From>
918   <wsa:MessageID>urn:uuid:f81d4fae-8dec-11d0-a765-
919   00a0c91e6bf6</wsa:messageID>
920   ...
921 </soap:Header>
922 <soap:Body>
923   ...
924 </soap:Body>
925 </soap:Envelope>
926
```

927

928

S3:

```
929 <soap:Envelope ...>
930 <soap:Header>
931   <wsa:To>http://example.com/callback</wsa:To>
932   <myNS:SomeID>1</myNS:SomeID>
933   <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
934   bindings/ws/callback">
935     urn:uuid:f81d4fae-8dec-11d0-a765-00a0c91e6bf6
936   </wsa:RelatesTo>
937   ...
938 </soap:Header>
939 <soap:Body>
940   ...
941 </soap:Body>
942 </soap:Envelope>
943
```

943

944

R3:

```
945 <soap:Envelope ...>
946 <soap:Header>
947   <wsa:From>
948     <wsa:Address>http://example.com/callback-other</wsa:Address>
949     <wsa:ReferenceProperties>
950       <myNS:SomeID>2</myNS:SomeID>
951     </wsa:ReferenceProperties>
952   </wsa:From>
953   <wsa:MessageID>urn:uuid:f81d4fae-9dec-11d0-a765-
954   00a0c91e6bf6</wsa:messageID>
955   ...
956 </soap:Header>
957 <soap:Body>
958   ...
959 </soap:Body>
960 </soap:Envelope>
```

963

964

S4, S5:

```
965 <soap:Envelope ...>
966 <soap:Header>
967   <wsa:To>http://example.com/callback-other</wsa:To>
968   <myNS:SomeID>2</myNS:SomeID>
969   <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
970   bindings/ws/callback">urn:uuid:f81d4fae-9dec-11d0-a765-
971   00a0c91e6bf6</wsa:RelatesTo>
972   ...
973 </soap:Header>
974 <soap:Body>
975   ...
976 </soap:Body>
977 </soap:Envelope>
```

978

979

E.1 Message Examples Using WS-MakeConnection

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

987

R4:

```
988 <soap:Envelope ...>
989 <soap:Header>
990 <wsa:From>
991 <wsa:Address>http://docs.oasis-open.org/ws-
992 rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-446655440010</wsa:Address>
993 </wsa:From>
994 <wsa:MessageID>urn:uuid:f81d4fae-10dec-11d0-a765-
995 00a0c91e6bf6</wsa:messageID>
996 ...
997 </soap:Header>
998 <soap:Body>
999 ...
1000 </soap:Body>
1001 </soap:Envelope>
```

1002

1003 **MakeConnection polling message (from R to S):**

```
1004 <soap:Envelope ...>
1005 <soap:Header>
1006 <wsa:Action>http://docs.oasis-open.org/ws-
1007 rx/wsmc/200702/MakeConnection</wsa:Action>
1008 ...
1009 </soap:Header>
1010 <soap:Body>
1011 <wsmc:MakeConnection>
1012 <wsmc:Address>http://docs.oasis-open.org/ws-
1013 rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-
1014 446655440010</wsmc:Address>
1015 </wsmc:MakeConnection>
1016 </soap:Body>
1017 </soap:Envelope>
```

1018

1019 **S6:**

```
1020 <soap:Envelope ...>
1021 <soap:Header>
1022 <wsa:To>http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id=650e8400-
1023 f29b-11d4-a716-446655440010</wsa:To>
1024 <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
1025 bindings/ws/callback">urn:uuid:f81d4fae-10dec-11d0-a765-
1026 00a0c91e6bf6</wsa:RelatesTo>
1027 ...
1028 </soap:Header>
1029 <soap:Body>
1030 ...
1031 </soap:Body>
1032 </soap:Envelope>
```

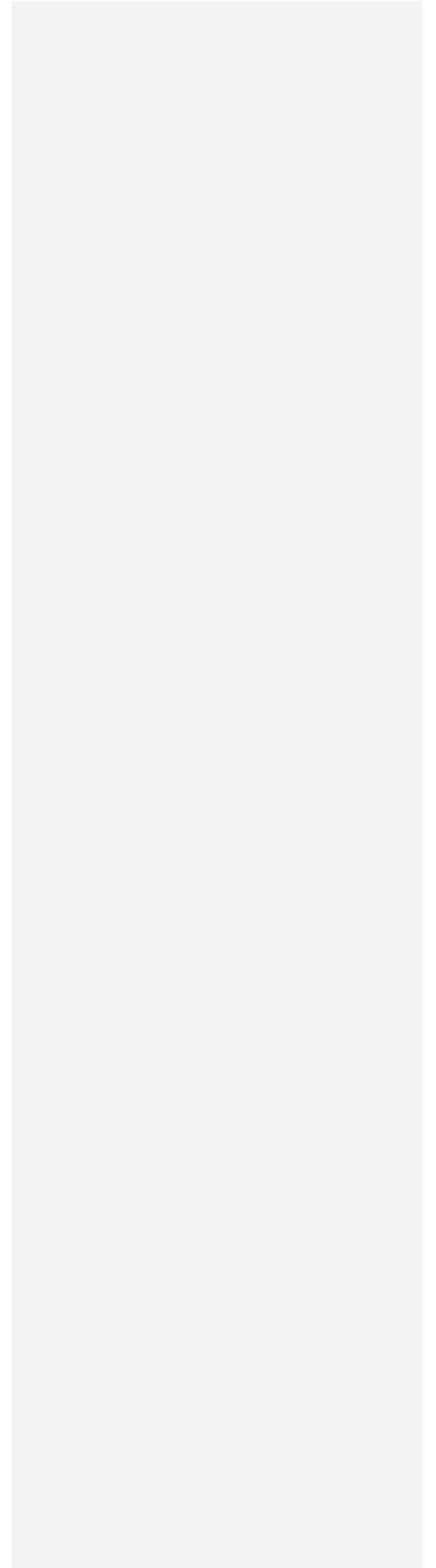
1033

F. Acknowledgements

1034 The following individuals have participated in the creation of this specification and are gratefully
1035 acknowledged:

1036 **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 Kavantzias	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.



1038

G. Revision History

1039 [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"> * Partially applied the resolution of issue 14 in the conformance section. * Applied resolution to issue 9. * Applied resolution to issue 15. * Applied resolution to issue 16. * Applied resolution to issue 10. * Applied resolution to issue 8. * Applied resolution to issue 3.
3	2008-06-12	Simon Holdsworth	<ul style="list-style-type: none"> * Completed application of resolution to issue 10 * Applied most of the editorial changes from Eric Johnson's review
4	2008-08-13	Anish Karmarkar	<ul style="list-style-type: none"> * Applied rest of Eric Johnson's ed review comments. * Applied resolution of issue 13. * Reapplied resolution of issue 15 (it was not applied correctly before) * Applied resolution of issue 19. * Applied resolution of issue 30. * Applied resolution of issue 32. * Applied resolution of issue 36. * Applied resolution of issue 38.
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"> Applied resolution of issue 11 Applied resolution of issue 49 Applied action item 20080904-1
cd02	2009-02-16	Simon Holdsworth	Renamed, applied editorial issues

cd02-rev1	2009-06-02	Anish Karmarkar	<ul style="list-style-type: none"> * Applied resolution of issue 61 by using the document at http://www.oasis-open.org/apps/org/workgroup/sca-bindings/download.php/32160/sca-binding-ws-1.1-spec-cd02-issue61-rev3.doc as the base document. * Updated NS URI (Applied action item 20090311-2). * Updated Copyright statement in various places. * Updated schema per http://lists.oasis-open.org/archives/sca-bindings/200903/msg00057.html (Applied action item 20090312-1). * Applied resolution of issue 23, 25, 43, 54, 55, 64. * Replaced 3 occurrences of 'required' with 'specified'. * Recreated all bookmarks, cross-references, and conformance item table.
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"> * Removed ':' from 40005, reformatted 40006/40007. * minor ed changes pointed out by SimonN. * minor formatting changes. * modified BWS20018 to remove the first sentence.
cd02-rev4	2009-06-17	Anish Karmarkar	<ul style="list-style-type: none"> * Not fixed in this rev, but issue 57 resolution was applied in previous rev. * Added list of participants in the Ack section. * Ed changes pointed out by Eric.
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)

1040