

Service Component Architecture Web Service Binding Specification Version 1.1

Committee Specification Draft 045 /
Public Review ~~02~~Draft 03

~~6 May 2010~~

28 July 2011

Specification URIs:

This ~~V~~ersion:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-csprd03.pdf>
(Authoritative)
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-csprd03.html>
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-csprd03.doc>

Previous ~~V~~ersion:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd04.pdf> (Authoritative)
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd04.html>
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-cd04.doc>

Latest ~~V~~ersion:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec.pdf> (Authoritative)
<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 (simon_holdsworth@uk.ibm.com, ~~IBM~~), IBM

Editor(s):

Editors:

Simon Holdsworth (simon_holdsworth@uk.ibm.com, ~~IBM~~), IBM
Anish Karmarkar (Anish.Karmarkar@oracle.com, ~~Oracle~~), Oracle

Related ~~W~~ork:

This specification replaces or supersedes:

- Service Component Architecture Web Service Binding Specification Version 1.00, March 21 2007
http://www.osea.org/download/attachments/35/SCA_WebServiceBinding_V100.pdf?version=2

This specification is related to:

- ~~OASIS Committee Draft 05~~, “Service Component Architecture Assembly Model Specification Version 1.1”, ~~January 2010~~

- [. Latest version.](http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec.html)
- [OASIS Committee Draft 02, "SCA Policy Framework Version 1.1. Latest version.](http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1.html) February 2009
- [TestCases for the SCA Web Service Binding Specification Version 1.1. Latest version.](http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-testcases.html)

Declared XML Namespace(s):namespace:

<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~~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 Citation format:

When referencing this specification is located at the following citation format should be used:

[SCA-WSBinding]

Service Component Architecture Web Service Binding Specification Version 1.1. 28 July 2011. OASIS Committee Specification Draft 05 / Public Review Draft 03.
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-spec-csprd03.html>

Notices

Copyright © OASIS~~© 2005, 2010.~~ Open 2011. 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~~ 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 names "OASIS"~~™~~, "SCA"~~™~~, and "Service Component Architecture"~~™~~ are trademarks of ~~OASIS~~ 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
1.5	Testcases.....	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.....	11
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.....	12
2.7	Support for WSDL extensibility.....	12
2.8	Intents listed in the bindingType.....	13
2.9	Intents and binding configuration.....	13
3	Web Service Binding Examples.....	14
3.1	Example Using WSDL documents.....	14
3.2	Examples Without a WSDL Document.....	14
4	Transport Binding.....	16
4.1	Intents.....	16
4.2	Default Transport Binding Rules.....	16
4.2.1	WS-I Basic Profile Alignment.....	16
4.2.2	Default Transport Binding Rules.....	16
5	Implementing SCA Callbacks using Web Services.....	18
5.1	SCA Web Services Callback Protocol.....	18
5.2	SCA Web Services Callback Protocol with WS-MakeConnection.....	19
5.3	Policy Assertion for SCA Web Services Callback Protocol.....	19
5.3.1	Assertion Model.....	19
5.3.2	Normative Outline.....	19
5.3.3	Assertion Attachment.....	20
5.3.4	Assertion Example.....	20
5.3.5	Security Considerations.....	21
6	Conformance.....	22
6.1	SCA WS Binding XML Document.....	22
6.2	Web Service Callback Service.....	22
6.3	Web Service Callback Client.....	22
6.4	SCA Runtime.....	22
A.	Web Services XML Binding Schema: sca-binding-ws-1.1.xsd (Normative).....	24
B.	SCA Web Services Callback Protocol Policy Assertion XML Schema: sca-binding-webservice-callback-1.1.xsd (Normative).....	25
C.	Conformance Items (Normative).....	26
D.	WSDL Generation (Non-Normative).....	29

E.	SCA Web Services Callback Protocol Message Examples (Non-Normative)	31
E.1	Message Examples Using WS-MakeConnection.....	33
F.	Acknowledgements (Non-Normative).....	35
G.	Revision History (Non-Normative).....	36

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
soap	Can be either "http://schemas.xmlsoap.org/soap/envelope/" or "http://www.w3.org/2003/05/soap-envelope"	Defined by SOAP 1.1 or SOAP 1.2
wsdl	"http://www.w3.org/ns/wsdl-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/200912"	Defined by the SCA specifications

Table 1-1: Prefixes and Namespaces Used in this Specification

28 1.2 Normative References

- 29 [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
30 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 31 [SCA-Assembly] OASIS Committee [Specification](#) Draft 057, “Service Component Architecture
32 *Assembly Model Specification Version 1.1*”, January 2010
33 [http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-
34 csd07.pdf](http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-csd07.pdf)
- 35 [SCA-Policy] OASIS Committee Draft 024, “SCA Policy Framework Specification Version 1.1”,
36 [February 2009, September 2010](#)
37 <http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd04.pdf>
- 38 [SCA-JCAA] OASIS Committee [Specification](#) Draft 03, “05, Service Component Architecture
39 *SCA Java-J Common Annotations and APIs Specification Version 1.1*”, May
40 [2009, November 2010](#)
41 <http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec-csd05.pdf>
- 42 [WSDL11] E. Christensen et al, *Web Service Description Language (WSDL) 1.1*,
43 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>, W3C Note, March 15 2001.
- 44 [WSDL20] Chinnici et al, *Web Service Description Language (WSDL) Version 2.0 Part 1:
45 Core Language*, <http://www.w3.org/TR/2007/REC-wsdl20-20070626/>, W3C
46 Recommendation, June 26 2007.
- 47 [WSI-Profiles] “Basic Profile Version 1.1”
48 <http://www.ws-i.org/Profiles/BasicProfile-1.1.html>,
49 “Attachments Profile Version 1.0”
50 <http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html>,
51 “Simple SOAP Binding Profile Version 1.0”
52 <http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html>,
53 “Basic Security Profile Version 1.0”
54 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>
- 55 [JAX-WS] “JSR 224: Java™ API for XML-Based Web Services (JAX-WS) 2.0”
56 <http://jcp.org/en/jsr/detail?id=224>
- 57 [SOAP11] Box et al, “Simple Object Access Protocol (SOAP) 1.1”
58 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>, W3C Note May 2000
- 59 [SOAP] Gudgin et al, “SOAP Version 1.2 Part 1: Messaging Framework”
60 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>, W3C
61 Recommendation June 2003; Box et al, “Simple Object Access Protocol (SOAP)
62 1.1” <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>, W3C Note May 2000
- 63 [SOAP12Adjuncts] Gudgin et al, “SOAP Version 1.2 Part 2: Adjuncts (Second Edition)”
64 <http://www.w3.org/TR/soap12-part2/>, W3C Recommendation April 2007
- 65 [WS-Addr] Gudgin et al, “Web Services Addressing 1.0 – Core”
66 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>, W3C
67 Recommendation May 2006
- 68 [W11-SOAP12] Angelov et al, “WSDL 1.1 Binding Extension for SOAP 1.2”
69 <http://www.w3.org/Submission/wsdl11soap12/>, W3C Member Submission April
70 2006
- 71 [WS-Addr-SOAP] Gudgin et al, “Web Services Addressing 1.0 – SOAP Binding”
72 <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>, W3C
73 Recommendation May 2006
- 74 [WS-MC] OASIS Standard “Web Services Make Connection (WS-MakeConnection)
75 Version 1.1”, February 2009
76 <http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.doc>
- 77 [WS-Policy] Vedamuthu et al, “Web Services Policy 1.5 – Framework”
78 <http://www.w3.org/TR/2007/REC-ws-policy-20070904/>, W3C Recommendation
79 September 2007

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

83 1.3 Non-Normative References

84 [WSI-AP] "Attachments Profile Version 1.0" [http://www.ws-](http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html)
85 [i.org/Profiles/AttachmentsProfile-1.0.html](http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html)
86 [MTOM] Gudgin et al, "SOAP Message Transmission Optimization Mechanism"
87 <http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/>, W3C
88 Recommendation January 2005
89 [WS-Security] Oasis Standard "Web Services Security: SOAP Message Security 1.1 (WS-
90 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)
91 [spec-os-SOAPMessageSecurity.pdf](http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)
92 [WS-Testcases] OASIS Committee Draft 01, "TestCases for the SCA Web Service
93 [Binding Specification Version 1.1"](http://docs.oasis-open.org/opencsa/sca-bindings/sca-wsbinding-1.1-testcases-cd01.pdf), July 2010, [open.org/opencsa/sca-bindings/sca-wsbinding-1.1-testcases-cd01.pdf](http://docs.oasis-
94 <a href=)

95 1.4 Naming Conventions

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

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

107 1.5 Testcases

108 [TestCases for SCA Web Service Binding Specification Version 1.1 \[WS-Testcases\]](#) defines test cases for
109 [this specification. The TestCases represent a series of tests that SCA runtimes are expected to pass in](#)
110 [order to claim conformance to the requirements of this specification.](#)

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 ... />?
  <wsa:EndpointReference>...</wsa: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. [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. [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. [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). [BWS20006]

– Port:

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

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. [BWS20007] The SCA runtime MUST expose an endpoint

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

163 – Binding:

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

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

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

176 When the *wsdl.binding* form of *wsdlElement* is used, the endpoint address URI for an SCA
177 reference MUST be specified by either the *@uri* attribute on the binding or a WS-Addressing
178 *wsa:EndpointReference* element, except where the SCA Assembly Model specification [SCA-
179 **Assembly**] states that the *@uri* attribute can be omitted. [BWS20015]

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

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

186 If the *@wsdli:wsdlLocation* attribute is used the *@wsdlElement* attribute MUST also be specified.
187 [BWS20017]

188 The value of the *@wsdli:wsdlLocation* attribute MUST identify an existing WSDL 1.1 document.
189 [BWS20018]

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

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

194 • **/binding.ws/wsa:EndpointReference** – when present this element provides the WS-Addressing
195 [WS-Addr] *wsa:EndpointReference* that specifies the endpoint for the service or reference.

196 A *binding.ws* element MUST NOT contain more than one of the following: the *@uri* attribute; the
197 *@wsdlElement* attribute referring to a WSDL port or to a WSDL service; the *wsa:EndpointReference*
198 element. [BWS20019]

199 The endpoint address URI for an SCA service or the callback element of an SCA reference is determined
200 as specified in section 2.2. For the *callback* element of an SCA service, the binding MUST NOT specify
201 an endpoint address URI or a WS-Addressing *wsa:EndpointReference*. [BWS20020]

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

204 The SCA runtime SHOULD support the element <wsa:EndpointReference>. [BWS20022] If an SCA
205 runtime does not support the element <wsa:EndpointReference>, then it MUST reject an SCA WS
206 Binding XML document (as defined in Section 5.1) that contains the element. [BWS20023]

207 The <binding.ws> element MUST conform to the XML schema defined in sca-binding-webservice-
208 1.1.xsd. [BWS20024]

209 2.1 Compatibility of SCA Service Interfaces and WSDL portTypes

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

- 212 1. The SCA service interface is remotable.
- 213 2. The operations on the portType are the same as the operations on the SCA service interface, with the
214 same operation name, same input types (taking order as significant), same output types (taking order
215 as significant), and same fault/exception types. If the SCA service interface is not a WSDL portType,
216 it is mapped to a WSDL portType for the purposes of this comparison. The mapping is defined in the
217 relevant SCA specification for the interface type. If the interface cannot be mapped to WSDL, the
218 SCA service interface is not compatible with the WSDL portType.
- 219 3. WSDL 1.1 message parts can point either to an XML Schema element declaration or to an XML
220 Schema type declaration. When determining compatibility between two WSDL operations, a
221 message part that points to an XML Schema element is considered to be incompatible with a
222 message part that points to an XML Schema type.
- 223 4. If either the portType or the SCA service interface declares an SCA callback interface, then both the
224 portType and the SCA service interface declare callback interfaces and these callback interfaces are
225 compatible according to points 1 through 3 above.

226 2.2 Endpoint URI resolution

227 This specification does not mandate any particular way to determine the URI for a web services binding
228 on an SCA service. An absolute URI can be indicated by the @uri attribute, by the URI in a wsa:Address
229 element within an wsa:EndpointReference element, or by the URI indicated in a WSDL port via a
230 @wsdlElement attribute. Implementations can use the specified URI as the service endpoint URI or they
231 can use a different URI which might include portions of the specified URI. For example, the service
232 endpoint URI might be produced by modifying any or all of the host name, the port number, and a portion
233 of the path.

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

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

- 239 A. The value of the @uri attribute
- 240 B. The value of the wsa:Address element of the wsa:EndpointReference element
- 241 C. The value of the address element of the WSDL port referenced by the @wsdlElement attribute
- 242 D. The value of the address element of one of the set of available WSDL ports as specified under the
243 definition of the @wsdlElement attribute when it references a WSDL service element

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

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

246 2.3 Interface mapping

247 When *binding.ws* is used on a service or reference with an interface that is not defined by *interface.wsdl*,
248 the SCA runtime MUST derive a WSDL portType for the service or reference from the interface using the
249 WSDL-mapping rules defined for that SCA interface type. [BWS20027]

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

252 For example, for *interface.java*, the mapping to a WSDL portType is as defined in the SCA Java Common
253 Annotations and API Specification **[SCA-JCAA]**.
254 *binding.ws* implementations can use appropriate standards, for example WS-I AP 1.0 **[WSI-AP]** or MTOM
255 **[MTOM]**, to map interface parameters to binary attachments transparently to the target component.

256 2.4 Production of WSDL description for an SCA service

257 ~~[BWS20029]~~
258 ~~[BWS20030]~~Any service hosted by an SCA runtime with one or more web service bindings with HTTP
259 endpoints is strongly encouraged to return a WSDL description of the service in response to an HTTP
260 GET request with the `?wsdl` suffix added to that HTTP endpoint URL. Regardless of the protocol
261 supported by an SCA service endpoint using the web services binding, the SCA runtime is strongly
262 encouraged to provide some means of obtaining the WSDL description of the service. This can include
263 out of band mechanisms, for example publication to a UDDI registry.

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

266 2.5 Additional binding configuration data

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

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

273 2.6 Web Service Binding and SOAP Intermediaries

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

276 2.7 Support for WSDL extensibility

277 When a *binding.ws* element uses the `@wsdlElement` attribute, the details of the binding are specified by
278 the WSDL element referenced by the value of the attribute. Per the WSDL specification, WSDL allows for
279 extensibility via elements as well as attributes, and it specifies rules for processing such elements. This
280 specification does not constrain the use of such extensibility in WSDL and relies on the rules specified in
281 the WSDL specification for processing such extended elements.

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

284 The SCA runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP **[WSDL11]**,
285 as identified by the WSDL element `wsoap11:binding` that has the `@transport` attribute with a value of
286 `"http://schemas.xmlsoap.org/soap/http"`. **[BWS20033]**

287 ~~[BWS20034]~~

288 The SCA runtime can support the WSDL 1.1 binding extension for SOAP 1.2 over HTTP **[W11-SOAP12]**,
289 as identified by the WSDL element `wsoap12:binding` that has the `@transport` attribute with a value of
290 `"http://schemas.xmlsoap.org/soap/http"`. Because a WSDL document might contain extension elements
291 that cannot be supported by the SCA runtime, when using the `@wsdlElement` form of *binding.ws* it is not
292 possible to determine whether the binding is supported by the SCA runtime without parsing the
293 referenced WSDL element and its dependent elements.

294 **2.8 Intents listed in the bindingType**

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

297 **2.9 Intents and binding configuration**

298 This binding mandates support for SOAP 1.1 and encourages SOAP 1.2 support. The <bindingType>
299 element associated with this binding MUST include the SOAP.v1_1 intent in its @mayProvides or
300 @alwaysProvides attributes. **[BWS20035]** ~~**[BWS20036]**~~ Where the binding implementation provides
301 support for SOAP 1.2 over HTTP described above, it is good practice for the <bindingType> element
302 associated with this binding to include the SOAP.v1_2 intent in its @mayProvides attribute. For more
303 details on the <bindingType> element see **[SCA-Policy]**.

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

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

308 3 Web Service Binding Examples

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

312 3.1 Example Using WSDL documents

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

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

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

```
322  
323 <?xml version="1.0" encoding="ASCII"?>  
324 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
325     name="MyValueComposite">  
326     <service name="MyValueService">  
327         <interface.java interface="services.myvalue.MyValueService"/>  
328         <binding.ws wsdlElement="http://www.example.org/MyValueService#  
329             wsdl.binding(MyValueService/MyValueServiceSOAP)"/>  
330         ...  
331     </service>  
332  
333     ...  
334  
335     <reference name="StockQuoteReference1">  
336         <interface.java interface="services.stockquote.StockQuoteService"/>  
337         <binding.ws wsdlElement="http://www.example.org/StockQuoteService#  
338             wsdl.service(StockQuoteService) "  
339             wsdl:wsdlLocation="http://www.example.org/StockQuoteService  
340                 http://www.example.org/StockQuoteService.wsdl"/>  
341     </reference>  
342  
343     <reference name="StockQuoteReference2">  
344         <interface.java interface="services.stockquote.StockQuoteService"/>  
345         <binding.ws wsdlElement="http://www.example.org/StockQuoteService#  
346             wsdl.binding(StockQuoteBinding) "  
347             wsdl:wsdlLocation="http://www.example.org/StockQuoteService  
348                 http://www.example.org/StockQuoteService.wsdl"  
349                 uri="http://www.example.org/StockQuoteService5"/>  
350     </reference>  
351 </composite>
```

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

353 3.2 Examples Without a WSDL Document

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

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

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

363

```
364 <?xml version="1.0" encoding="ASCII"?>
365 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
366     name="MyValueComposite">
367
368     <service name="MyValueService">
369         <interface.java interface="services.myvalue.MyValueService"/>
370         <binding.ws/>
371         ...
372     </service>
373
374     ...
375
376     <reference name="StockQuoteService">
377         <interface.java interface="services.stockquote.StockQuoteService"/>
378         <binding.ws uri="http://www.example.org/StockQuoteService"/>
379     </reference>
380 </composite>
```

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

382

383 Snippet 3-3 shows the use of the binding element without a WSDL document, with multiple SOAP
384 bindings with non-default values. The SOAP 1.2 binding name defaults to the service name, the SOAP
385 1.1 binding is given an explicit name. The reference has a web service binding which uses SOAP 1.2,
386 but otherwise uses all the defaults for SOAP binding. The reference binding name defaults to the
387 reference name.

388

```
389 <?xml version="1.0" encoding="ASCII"?>
390 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
391     name="MyValueComposite">
392
393     <service name="MyValueService">
394         <interface.java interface="services.myvalue.MyValueService"/>
395         <binding.ws name="MyValueServiceSOAP11" requires="SOAP.v1_1"/>
396         <binding.ws requires="SOAP.v1_2"/>
397         ...
398     </service>
399
400     ...
401
402     <reference name="StockQuoteService">
403         <interface.java interface="services.stockquote.StockQuoteService"/>
404         <binding.ws uri="http://www.example.org/StockQuoteService"
405             requires="SOAP.v1_2"/>
406     </reference>
407 </composite>
```

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

409 4 Transport Binding

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

415 4.1 Intents

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

- 418 • SOAP

419 When the SOAP intent is required, the SCA runtime MUST transmit and receive messages using
420 SOAP. One or more SOAP versions can be used. [BWS40001]

- 421 • SOAP.v1_1

422 When the SOAP.v1_1 intent is required, the SCA runtime MUST transmit and receive messages
423 using only SOAP 1.1. [BWS40002]

- 424 • SOAP.v1_2

425 When the SOAP.v1_2 intent is required, the SCA runtime MUST transmit and receive messages
426 using only SOAP 1.2. [BWS40003]

427 4.2 Default Transport Binding Rules

428 4.2.1 WS-I Basic Profile Alignment

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

- 432 • that every message part references an XML Schema type (rpc-literal pattern)
- 433 • or that every message references exactly zero or one XML Schema elements (document-literal
434 pattern)

435 For an SCA service or reference element, the portType from the service's or reference's interface or
436 derived from that interface MUST follow either the rpc-literal pattern or the document-literal pattern.
437 [BWS40004]

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

440 4.2.2 Default Transport Binding Rules

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

- 442 • HTTP-based transfer protocol;
- 443 • SOAP 1.1 binding;
- 444 • "literal" format as described in section 3.5 of [WSDL11];
- 445 • Either the document literal or rpc literal pattern, depending on the service or reference interface as
446 described in section 4.2.1;
 - 447 – For document literal pattern, each message uses "document" style, as per section 3.5 of
448 [WSDL11];

- 449 – For rpc-literal pattern, each message uses "rpc" style, as per section 3.5 of **[WSDL11]** and the
450 child elements of the SOAP Body element are namespace qualified with a non-empty namespace
451 name;
- 452 • For SOAP 1.1 messages, the SOAPAction HTTP header described in section 6.1.1 of **[SOAP11]**
453 represents the empty string, in quotes ("");
 - 454 • For SOAP 1.2 messages, the SOAP Action feature described in section 6.5 of **[SOAP12Adjuncts]**
455 does not appear;
 - 456 • All WSDL message parts are carried in the SOAP body.
- 457 In the event that the transport details are not determined by use of the @wsdlElement attribute, @uri
458 attribute, wsa:EndpointReference element, policy intents, policy sets or extensions to the binding.ws
459 element, an SCA runtime **MUST** enable the default transport binding rules. **[BWS40005]**
- 460 When using the default transport binding rules, the SCA runtime can provide additional WSDL bindings,
461 unless policy is applied that explicitly restricts this.
- 462 | When using the default transport binding rules with the rpc-literal pattern, the SCA runtime **MUST** use the
463 structural URI associated with the binding as the namespace of the child elements of the SOAP body
464 element. **[BWS40007]**

465 5 Implementing SCA Callbacks using Web Services

466 5.1 SCA Web Services Callback Protocol

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

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

- 475 1. Every request message from the WSCB Client that invokes the forward interface MUST contain a
476 Callback EPR. **[BWS50002]** If the request message contains the `wsa:From` SOAP header block
477 then the `wsa:From` header block specifies the Callback EPR. If the `wsa:From` header block is not
478 present then the `wsa:ReplyTo` header block specifies the Callback EPR.
479 If the Callback EPR's [address] value is
480 "`http://www.w3.org/2005/08/addressing/anonymous`" or
481 "`http://www.w3.org/2005/08/addressing/none`" then the WSCB Service MUST generate
482 the Invalid Addressing Header fault as specified in Section 6.4.1 of **[WS-Addr-SOAP]**. **[BWS50004]**
483 Such a fault can include additional [Subsubcode]
484 `wsa:OnlyNonAnonymousAddressSupported`.
- 485 2. A request message that invokes the forward interface can contain the `wsa:MessageID` SOAP
486 header block. If there is a need to have the callback request message correlated to an individual
487 forward request message, the `wsa:MessageID` SOAP header block can be used for this purpose.
- 488 3. When the WSCB Service invokes the callback interface, it MUST use the Callback EPR from a
489 request message that invoked the forward interface. **[BWS50005]** Once the Callback EPR is
490 selected, the WSCB Service MUST follow the rules defined in Section 3.3 of **[WS-Addr]** to invoke
491 operations on the callback interface. **[BWS50006]**

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

499 If the request message from which the Callback EPR was obtained did not contain the `wsa:MessageID`
500 SOAP header block, the WSCB Service MUST NOT include a `wsa:RelatesTo` SOAP header block with
501 a relationship type value of "`http://docs.oasis-open.org/opencsa/sca-
502 bindings/ws/callback`" in the callback message. **[BWS50009]**

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

510 5.2 SCA Web Services Callback Protocol with WS-MakeConnection

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

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

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

524 The unique-String value is a globally unique value such as a UUID, as defined by the WS-
525 MakeConnection specification.

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

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

538 5.3 Policy Assertion for SCA Web Services Callback Protocol

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

544 5.3.1 Assertion Model

545 The WSCallback policy assertion indicates that the WSCB Client and the WSCB Service **MUST use SCA**
546 **Web Services Callback Protocol to implement callbacks.** [**BWS50010**] Specifically, the protocol
547 determines the requirements on the forward request message, the EPR used for callbacks and the
548 requirements on the callback request message.

549 5.3.2 Normative Outline

550 The normative outline for the WSCallback assertion is:

551

```
552 <sca:WSCallback ...>  
553   ...  
554 </sca:WSCallback>
```

555 *Snippet 5-1: WSCallback Assertion*

556

557 The content model of the WSCallback element is:

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

560 5.3.3 Assertion Attachment

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

- 562 • Endpoint Policy Subject

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

566 ~~[BWS50013]~~

567 The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects allowed for a
568 WSCallback policy assertion but which cannot have WSCallback policy assertions attached:
569 wsdlPortType

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

- 572 • `wsdl:port`
- 573 • `wsdl:binding`

574 5.3.4 Assertion Example

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

576

```
577 (01) <wsdl:definitions  
578 (02)   targetNamespace="example.com"  
579 (03)   xmlns:tns="example.com"  
580 (04)   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"  
581 (05)   xmlns:wsp="http://www.w3.org/ns/ws-policy"  
582 (06)   xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
583 (07)   xmlns:wssu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
584 wssecurity-utility-1.0.xsd">  
585 (08)  
586 (09) <wsp:UsingPolicy wsdl:required="true" />  
587 (10)  
588 (11) <wsp:Policy wsu:Id="MyPolicy" >  
589 (12)   <sca:WSCallback/>  
590 (13) </wsp:Policy>  
591 (14)  
592 (15) <!-- omitted elements -->  
593 (16)  
594 (17) <wsdl:binding name="MyBinding" type="tns:MyPortType" >  
595 (18)   <wsp:PolicyReference URI="#MyPolicy" />  
596 (19)   <!-- omitted elements -->  
597 (20) </wsdl:binding>  
598 (21)  
599 (22) </wsdl:definitions>
```

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

601

602 Line (09) in Snippet 5-2 indicates that WS-Policy is in use as a required extension. Lines (11-13) are a
603 policy expression that includes a WSCallback policy assertion (line 12) to indicate that SCA Web Services
604 Callback protocol is used. Lines (17-20) are a WSDL binding. Line (18) indicates that the policy in lines

605 (11-13) applies to this binding, specifically indicating that SCA Web Services Callback protocol is used
606 over all the messages in the binding.

607 **5.3.5 Security Considerations**

608 ~~–[BWS50014]–[BWS50015]~~It is good practice for Policies and Assertions to be signed to prevent
609 tampering. It is acceptable for an SCA runtime to reject a Policy that is not signed or where there is no
610 associated security token which indicates that the signer has appropriate claims for the policy. That is, a
611 relying party shouldn't rely on a policy unless the policy is signed and presented with sufficient claims to
612 pass the relying parties acceptance criteria.

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

616 6 Conformance

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

620 This specification defines four targets for conformance:

- 621 a) SCA WS Binding XML Document
- 622 b) Web Service Callback Service (WSCB Service)
- 623 c) Web Service Callback Client (WSCB Client)
- 624 d) SCA Runtime

625 6.1 SCA WS Binding XML Document

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

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

633 A WSDL 1.1 portType element associated with an SCA service or reference **MUST NOT** have the
634 WSCallback policy assertion attached. "associated with" is intended to cover both the referencing of a
635 concrete WSDL document from within an SCA XML document (by any element) and also cover the
636 dynamic creation and advertising of a WSDL by a runtime service.

637 6.2 Web Service Callback Service

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

641 6.3 Web Service Callback Client

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

645 6.4 SCA Runtime

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

- 648 1. The implementation **MUST** comply with all statements in Appendix C: Conformance Items related to
649 an SCA Runtime, except for those that originate from Section 5, notably all "MUST" statements have
650 to be implemented.
- 651 ~~2. The implementation **MAY** support the SCA Web Services Callback Protocol. If it does, it **MUST** be a~~
652 ~~compliant WSCB Service and WSCB Client.~~
- 653 ~~3. The implementation **MAY** support the SCA Web Services Callback Protocol in conjunction with WS-~~
654 ~~MakeConnection. If it does, it **MUST** be a compliant WSCB Service, WSCB Client, and it **MUST**~~
655 ~~comply with the requirements of WS-MakeConnection.~~
- 656 ~~4.2.~~ The implementation **MUST** conform to the SCA Assembly Model Specification Version 1.1 **[SCA-**
657 **Assembly]**, and to the SCA Policy Framework Version 1.1 **[SCA-Policy]**.

658 | ~~5.3.~~ The implementation MUST reject a SCA WS Binding XML Document that is not conformant per
659 Section 6.1.

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

664
665

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

```
666 <?xml version="1.0" encoding="UTF-8"?>
667 <!-- Copyright (C) OASIS (R) 2005,2010. All Rights Reserved.
668     OASIS trademark, IPR and other policies apply. -->
669 <schema xmlns="http://www.w3.org/2001/XMLSchema"
670     targetNamespace="http://docs.oasis-open.org/ns/opencsa/sca/200912"
671     xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200912"
672     xmlns:wsdli="http://www.w3.org/ns/wsdli-instance"
673     xmlns:wsa="http://www.w3.org/2005/08/addressing"
674     elementFormDefault="qualified">
675
676     <import namespace="http://www.w3.org/ns/wsdli-instance"
677         schemaLocation="http://www.w3.org/2007/05/wsdli/wsdli20-
678 instance.xsd"/>
679     <import namespace="http://www.w3.org/2005/08/addressing"
680         schemaLocation="http://www.w3.org/2006/03/addressing/ws-
681 addr.xsd"/>
682
683     <include schemaLocation="sca-core-1.1-cd05.xsd"/>
684
685     <element name="binding.ws" type="sca:WebServiceBinding"
686         substitutionGroup="sca:binding"/>
687
688     <complexType name="WebServiceBinding">
689         <complexContent>
690             <extension base="sca:Binding">
691                 <sequence>
692                     <element ref="wsa:EndpointReference"
693                         minOccurs="0" maxOccurs="unbounded"/>
694                     <element ref="sca:extensions" minOccurs="0" maxOccurs="1"
695 />
696                 </sequence>
697                 <attribute name="wsdlElement" type="anyURI" use="optional"/>
698                 <attribute ref="wsdli:wsdlLocation" use="optional"/>
699             </extension>
700         </complexContent>
701     </complexType>
702 </schema>
```


703
704
705

706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724

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

```
<?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>
```

C. Conformance Items (Normative)

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

Conformance ID	Description
[BWS20001]	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 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]	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]	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], 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], and the WSDL binding MUST satisfy all the policy constraints of the binding.
[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>wsa:EndpointReference</i> element, except where the SCA Assembly Model specification [SCA-Assembly] states that the <i>@uri</i> attribute can be omitted.
[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>wsa: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>wsa: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><wsa:EndpointReference></i> .
[BWS20023]	If an SCA runtime does not support the element <i><wsa: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-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 portType 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 portType.
[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).
[BWS20033]	The SCA runtime MUST support the WSDL 1.1 binding extension for SOAP 1.1 over HTTP [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], 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.
[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, wsa: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 [WS-Addr-SOAP].
[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 [WS-Addr] to invoke operations on the callback interface.
[BWS50007]	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.

[BWS50008]	The <code>wsa:RelatesTo</code> 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 <code>wsa:MessageID</code> of the message from which the Callback EPR was obtained.
[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 "http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback" 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.

727

D. WSDL Generation (Non-Normative)

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

- 733 • component name = the value of the @name attribute of the component element containing the
734 binding.ws element
- 735 • service name = the value of the @name attribute of the service element containing the binding.ws
736 element
- 737 • binding name = the value of @name attribute of the binding.ws element, or the default if no @name
738 attribute is present
- 739 • SOAP version = either "SOAP11" or "SOAP12" as appropriate

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

- 741 • wsdl:definitions/@name = <component name> + "." + <service name>
- 742 • wsdl:definitions/@targetNamespace = <structural URI for the service>
- 743 • import each WSDL 1.1 portType, rather than putting them inline
- 744 • wsdl:binding/@name = <binding name> + <SOAP version> + "Binding"
- 745 • wsdl:service/@name = <service name>
- 746 • wsdl:port/@name = <binding name> + <SOAP version> + "Port"

747
748

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

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

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

- 754 1. R invokes the forward operation and sets the callback address to **RC1**. Let's call the message that
755 invokes the forward operation R1. S then calls the callback operation twice. Let's call the callback
756 messages S1 and S2
- 757 2. R invokes the forward operation again with the same callback address **RC1**. Let's call the message
758 that invokes the forward operation R2. S then calls the callback operation once. Let's call the callback
759 message S3.
- 760 3. R invokes the forward operation yet another time, but this time uses a difference callback address:
761 **RC2**. Let's call the message that invokes the forward operation R3. S then calls the callback
762 operation twice. Let's call the callback messages S4 and S5.

763 The messages R1, R2, R3, S1, S2, S3, S4 and S4 are shown. The namespace prefix 'soap' can be
764 bound to either the SOAP 1.1 or SOAP 1.2 namespace. The 'wsa' prefix is bound to the WS-Addressing
765 1.0 namespace.

766

767 **R1:**

```
768 <soap:Envelope ...>  
769 <soap:Header>  
770 <wsa:From>  
771 <wsa:Address>http://example.com/callback</wsa:Address>  
772 <wsa:ReferenceProperties>  
773 <myNS:SomeID>1</myNS:SomeID>  
774 </wsa:ReferenceProperties>  
775 </wsa:From>  
776 <wsa:MessageID>urn:uuid:f81d4fae-7dec-11d0-a765-  
777 00a0c91e6bf6</wsa:messageID>  
778 ...  
779 </soap:Header>  
780 <soap:Body>  
781 ...  
782 </soap:Body>  
783 </soap:Envelope>
```

784

785 **S1, S2:**

```
786 <soap:Envelope ...>  
787 <soap:Header>  
788 <wsa:To>http://example.com/callback</wsa:To>  
789 <myNS:SomeID>1</myNS:SomeID>  
790 <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-  
791 bindings/ws/callback">urn:uuid:f81d4fae-7dec-11d0-a765-  
792 00a0c91e6bf6</wsa:RelatesTo>  
793 ...  
794 </soap:Header>  
795 <soap:Body>  
796 ...  
797 </soap:Body>  
798 </soap:Envelope>
```

799

800 **R2:**

```
801 <soap:Envelope ...>
802   <soap:Header>
803     <wsa:From>
804       <wsa:Address>http://example.com/callback</wsa:Address>
805       <wsa:ReferenceProperties>
806         <myNS:SomeID>1</myNS:SomeID>
807       </wsa:ReferenceProperties>
808     </wsa:From>
809     <wsa:MessageID>urn:uuid:f81d4fae-8dec-11d0-a765-00a0c91e6bf6</wsa:messageID>
810   ...
811 </soap:Header>
812 <soap:Body>
813   ...
814 </soap:Body>
815 </soap:Envelope>
```

817

818 **S3:**

```
819 <soap:Envelope ...>
820   <soap:Header>
821     <wsa:To>http://example.com/callback</wsa:To>
822     <myNS:SomeID>1</myNS:SomeID>
823     <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
824 bindings/ws/callback">
825       urn:uuid:f81d4fae-8dec-11d0-a765-00a0c91e6bf6
826     </wsa:RelatesTo>
827   ...
828 </soap:Header>
829 <soap:Body>
830   ...
831 </soap:Body>
832 </soap:Envelope>
```

833

834 **R3:**

```
835 <soap:Envelope ...>
836   <soap:Header>
837     <wsa:From>
838       <wsa:Address>http://example.com/callback-other</wsa:Address>
839       <wsa:ReferenceProperties>
840         <myNS:SomeID>2</myNS:SomeID>
841       </wsa:ReferenceProperties>
842     </wsa:From>
843     <wsa:MessageID>urn:uuid:f81d4fae-9dec-11d0-a765-00a0c91e6bf6</wsa:messageID>
844   ...
845 </soap:Header>
846 <soap:Body>
847   ...
848 </soap:Body>
849 </soap:Envelope>
```

851

852 **S4, S5:**


```

853 <soap:Envelope ...>
854   <soap:Header>
855     <wsa:To>http://example.com/callback-other</wsa:To>
856     <myNS:SomeID>2</myNS:SomeID>
857     <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
858     bindings/ws/callback">urn:uuid:f81d4fae-9dec-11d0-a765-
859     00a0c91e6bf6</wsa:RelatesTo>
860     ...
861   </soap:Header>
862   <soap:Body>
863     ...
864   </soap:Body>
865 </soap:Envelope>

```

866 **E.1 Message Examples Using WS-MakeConnection**

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

873

874 **R4:**

```

875 <soap:Envelope ...>
876   <soap:Header>
877     <wsa:From>
878       <wsa:Address>http://docs.oasis-open.org/ws-
879       rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-446655440010</wsa:Address>
880     </wsa:From>
881     <wsa:MessageID>urn:uuid:f81d4fae-10dec-11d0-a765-
882     00a0c91e6bf6</wsa:messageID>
883     ...
884   </soap:Header>
885   <soap:Body>
886     ...
887   </soap:Body>
888 </soap:Envelope>

```

889

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

```

891 <soap:Envelope ...>
892   <soap:Header>
893     <wsa:Action>http://docs.oasis-open.org/ws-
894     rx/wsmc/200702/MakeConnection</wsa:Action>
895     ...
896   </soap:Header>
897   <soap:Body>
898     <wsmc:MakeConnection>
899       <wsmc:Address>http://docs.oasis-open.org/ws-
900       rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-
901       446655440010</wsmc:Address>
902     </wsmc:MakeConnection>
903   </soap:Body>
904 </soap:Envelope>

```

905

906 **S6:**

```
907 <soap:Envelope ...>
908   <soap:Header>
909     <wsa:To>http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id=650e8400-
910     f29b-11d4-a716-446655440010</wsa:To>
911     <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
912     bindings/ws/callback">urn:uuid:f81d4fae-10dec-11d0-a765-
913     00a0c91e6bf6</wsa:RelatesTo>
914     ...
915   </soap:Header>
916   <soap:Body>
917     ...
918   </soap:Body>
919 </soap:Envelope>
```

920

F. Acknowledgements (Non-Normative)

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

923 **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	TIBCO 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 Mischkinisky	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)

925 [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)
cd02-rev6	2009-06-24	Anish Karmarkar	<ul style="list-style-type: none"> * Applied resolution of issue 76, 79, 82. * Some very minor ed changes. * Reverted the document naming scheme to the old scheme.
cd02-rev7	2009-07-01	Simon Holdsworth	<ul style="list-style-type: none"> * Applied resolution of issue 2 * Fixed application of resolution of issue 76
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"> * Updated 'Notices' section for trademarks * Applied resolution of issue 99 points 9, 10, 16 * Added references per http://lists.oasis-open.org/archives/sca-bindings/200912/msg00013.html * Applied resolution of issue 84, 86, 91, 92, 116, 117, 118, 119 * Updated NS URI from 200903 to 200912
cd03-rev3	2010-03-31	Anish Karmarkar	<ul style="list-style-type: none"> * Updated schema appendix title to include "1.1" * Applied resolution of issue 124 * Ed changes associated with issue 124 resolution
cd03-rev4	2010-04-22	Anish Karmarkar	<ul style="list-style-type: none"> * Fixed ed issues pointed out at http://lists.oasis-open.org/archives/sca-bindings/201004/msg00004.html
cd03-rev5	2010-05-06	Anish Karmarkar	<ul style="list-style-type: none"> * Updated reference to assembly * Minor ed tweaks in the namespace prefix table * Applied resolution of issue 127
cd04	2010-05-14	Simon Holdsworth	Fix up for publication
wd041	2011-06-23	Mike Edwards	<ul style="list-style-type: none"> Issue 145 - Section 1.3, new section 1,5 Issue 160 - Section 5.3.3, section 6.1 Issue 161 - Section 5.3.5 Issue 163 - Section 2.4 Issue 164 - Section 4.2.2 Issue 165 - Section 5.3.5 Issue 166 - Section 6.4 Issue 168 - Sections 2.7, 2.9
wd042	2011-07-04	Simon Holdsworth	Editorial formatting changes
wd043	2011-07-22	Simon Holdsworth	Updated references to SCA Assembly, Policy and JCAA specifications