Service Component Architecture Web Service Binding Specification Version 1.1

Committee Draft 02 Issue 25 Resolution + Issue 2 Proposal v2

16th February, 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
Khanderao Kand, Oracle
Anish Karmarkar, Oracle
Sanjay Patil, SAP
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/200712

Abstract:

The SCA Web Service binding specified in this document applies to the services and references of an SCA composites. 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

Table of Contents

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.

The Web Service binding can point to an existing WSDL [WSDL] 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.

Comment [ask1]: Page: 5
Mentioning portType here does not seem appropriate.

Comment [ask2]: Page: 5
Dave would like to get rid of this.

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.

This specification also defines a protocol for implementing callbacks using the WS-Addressing Message Addressing Properties [WS-Addr].

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/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 [WSDL11-SOAP12]
sca	"http://docs.oasis-open.org/ns/opencsa/sca/200712"	Defined by the SCA specifications

1.2 Normative References

[RFC2119]	S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.
[SCA-Assembly]	http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec.pdf
[SCA-Policy]	http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec.pdf
[SCA-JCAA]	http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec.pdf
[WSDL11]	E. Christensen et al, *Web Service Description Language (WSDL) 1.1*, http://www.w3.org/TR/2001/NOTE-wsdl-20010315, W3C Note, March 15 2001.
[WSDL]	E. Christensen et al, *Web Service Description Language (WSDL) 1.1*, http://www.w3.org/TR/2001/NOTE-wsdl-20010315, W3C Note, March 15 2001.
	R. 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.
[WSI-Profiles]	http://www.ws-i.org/Profiles/BasicProfile-1.1.html
	http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html
	http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html
	http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html
[JAX-WS]	http://jcp.org/en/jsr/detail?id=224
[SOAP]	http://www.w3.org/TR/2003/REC-soap12-part1-20030624/
	http://www.w3.org/TR/2000/NOTE-SOAP-20000508/
[SOAP12Adjuncts]	SOAP Version 1.2 Part 2: Adjuncts (Second Edition) http://www.w3.org/TR/soap12-part2/
[WS-Addr]	http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/
[WS-Addr-SOAP]	http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/
[WSDL11-SOAP12]	http://www.w3.org/Submission/wsdl11soap12/
[WS-MC]	http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.html
[WS-Policy]	http://www.w3.org/TR/2007/REC-ws-policy-20070904
[WS-PA]	http://www.w3.org/TR/2007/REC-ws-policy-attach-20070904

1.3 Non-Normative References

[WSI-AP]	http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html
[MTOM]	http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/
[WS-RM]	http://docs.oasis-open.org/ws-rx/wsrm/200702/wsrm-1.2-spec-cd-01.html
[WS-Security]	http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

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"?
        wsdli:wsdlLocation="list of xs:anyURI pairs"?
        ...>
  <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 2.1 below describes how this attribute is interpreted.
- ***/binding.ws/@wsdlElement*** – when present this attribute specifies the URI of a WSDL element. This attribute points to the specified element in an existing WSDL document. The URI can have the following forms:
    - Service:

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

      In this case, the SCA runtime MUST make all the ports in the WSDL Service that have equivalent portTypes with the SCA service or reference available to the SCA service or reference.
    - Port (WSDL 1.1):

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

      In this case, the port in the WSDL 1.1 Service identified by the <binding.ws> element MUST implement a portType that is equivalent to the one specified for the SCA service or reference. The identified port MUST be made available to the SCA service or reference by the SCA runtime.
    - Endpoint (WSDL 2.0):

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

      In this case, the endpoint in the WSDL 2.0 Service identified by the <binding.ws> element MUST have an equivalent portType with the SCA service or reference. The identified endpoint MUST be made available to the SCA service or reference by the SCA runtime.

    - Binding:

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

      In this case, the WSDL binding identified by the <binding.ws> element MUST implement a portType that is equivalent to the one specified for the SCA service or reference. The SCA runtime MUST make the service or reference available via the specified WSDL binding. In this case, the endpoint address URI for an SCA reference MUST be specified by either the *@uri* attribute on the binding or a WS-Addressing *EndpointReference* element, except where the SCA Assembly specification states that the *@uri* attribute can be omitted. The endpoint address URI for an SCA service or the callback element of an SCA reference is determined as specified in section 2.1. For the *callback* element of an SCA service, the binding MUST NOT specify an endpoint address URI or a WS-Addressing EndpointReference..

- ***/binding.ws/@wsdli:wsdlLocation*** – when present this attribute specifies the location(s) of the WSDL document(s) associated with specific namespace(s). This attribute MAY be specified by the binding in the event that the <WSDL-namespace-URI> 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 <WSDL-namespace-URI>. The use of this attribute indicates that the WSDL binding points to an existing WSDL document. The semantics of this attribute are specified in Section 7.1 of WSDL 2.0 [WSDL].
- ***/binding.ws/endpointReference*** – when present this element provides the WS-Addressing [WS-Addr] EndpointReference that specifies the endpoint for the service or reference. When this element is present along with the *@wsdlElement* attribute on the parent element, the *@wsdlElement* attribute value MUST be of the 'Binding' form as specified above, i.e. <WSDL-namespace-URI>#wsdl.binding(<binding-name>).
- ***/binding.ws/@{any}*** - this is an extensibility mechanism to allow extensibility via attributes.
- ***/binding.ws/any*** – this is an extensibility mechanism to allow extensibility via elements.

The SCA runtime MUST support all the attributes of the <bind~~n~~ing.ws> element, namely @name, @uri, @requires, @policySets @wsdlElement, and @wsdli:wsdlLocation.

The SCA runtime SHOULD support the element <endpointReference>. If an SCA runtime does not support the element <endpointReference>, then it MUST reject an SCA WS Binding XML document (as defined in Section 6~~5~~.1) that contains the element.

The <binding.ws> element MUST conform to the XML schema defined in sca-binding-webservice.xsd.

2.1 Endpoint URI resolution

The rules for resolving the URI at which an SCA service is hosted, or SCA reference targets, when used with binding.ws (in precedence order) are:

1. The URIs in the endpoint(s) of the referenced WSDL
   or
   The URI specified by the *wsa:Address* element of the e*ndpointReference*,
2. The explicitly stated URI in the *@uri* attribute of the *binding.ws* element, which can be relative,
3. The structural URI as defined by the Assembly specification

An SCA runtime MUST follow rules listed above in determining the URI at which an SCA service is hosted or an SCA reference is targeted.

The URI in the WSDL endpoint or in the *wsa:Address* of an EPR MAY be a relative URI, in which case it is relative to the URI defined in (2) or (3). The *wsa:Address* element MAY be the empty relative URI, in which case it uses the URI defined in (2) or (3) directly. This enables the EPR writer to specify reference parameters, metadata and other EPR contents while letting the deployer choose the URI.

To reference a WSDL document and also specify an EPR, the *@wsdlElement* attribute MUST refer to a binding element in the WSDL.

2.2 Interface mapping

When *binding.ws* is used on a service or reference with an interface that is not defined by *interface.wsdl*, then a WSDL portType for the service or reference is derived from the interface by the rules defined for that SCA interface type. An SCA runtime MUST raise an error if the interface does not map to a WSDL portType.

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

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

2.3 Production of WSDL description for an SCA service

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. If none of the web service bindings have HTTP endpoints, then some other means of obtaining the WSDL description of the service SHOULD be provided by the SCA runtime. This can include out of band mechanisms, for example publication to a UDDI registry.

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

2.4 Additional binding configuration data

SCA runtime implementations MAY provide additional metadata that is associated with a web service binding, for example to enable JAX-WS [JAX-WS] handlers to be executed as part of the target component dispatch. The specification of such metadata is SCA runtime-specific and is outside of the scope of this document.

2.5 Web Service Binding and SOAP Intermediaries

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

2.6 Support for WSDL extensibility

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

This specification requires that an SCA runtime MUST support the WSDL extensions defined in the namespace associated with the prefix "sca" (as defined in section 1.1).

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".

The SCA runtime SHOULD support the WSDL 1.1 binding extension for SOAP 1.2 over HTTP [WSDL11-SOAP12], as identified by the WSDL element wsoap12:binding that has the @transport attribute with a value of "http://schemas.xmlsoap.org/soap/http".

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

2.7 Intents listed in the bindingType

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

2.8 Intents and binding configuration

This binding mandates support for SOAP 1.1 and encourages SOAP 1.2 support. The <bindingType> element associated with this binding MUST include the SOAP.1_1 intent in its @mayProvides or @alwaysProvides attributes. If the <bindingType> element associated with this binding does not include the SOAP.1_1 intent in its @alwaysProvides attribute, then the <bindingType> element SHOULD include the SOAP.1_2 intent in its @mayProvides attribute. For more details on the <bindingType> element see [SCA-Policy].

The SCA runtime MUST raise an error if the web service binding is configured with a policy intent(s) that conflicts with a binding instance's configuration. For example, it is an error to use the SOAP policy intent in combination with a WSDL binding that does not use SOAP.

3 Web Service Binding Examples

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

3.1 Example Using WSDL documents

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

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

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

```
<?xml version="1.0" encoding="ASCII"?>
<composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200712"
        name="MyValueComposite">
  <service name="MyValueService">
    <interface.java interface="services.myvalue.MyValueService"/>
    <binding.ws wsdlElement="http://www.example.org/MyValueService#
                     wsdl.endpoint(MyValueService/MyValueServiceSOAP)"/>
      ...
  </service>

  ...

  <reference name="StockQuoteReference1">
    <interface.java interface="services.stockquote.StockQuoteService"/>
    <binding.ws wsdlElement="http://www.example.org/StockQuoteService#
                     wsdl.service(StockQuoteService)"
    wsdli:wsdlLocation="http://www.example.org/StockQuoteService
                  http://www.example.org/StockQuoteService.wsdl"/>
  </reference>

  <reference name="StockQuoteReference2">
    <interface.java interface="services.stockquote.StockQuoteService"/>
    <binding.ws wsdlElement="http://www.example.org/StockQuoteService#
                     wsdl.binding(StockQuoteBinding)"
    wsdli:wsdlLocation="http://www.example.org/StockQuoteService
                  http://www.example.org/StockQuoteService.wsdl"
               uri="http://www.example.org/StockQuoteService5"/>
  </reference>
</composite>
```

3.2 Examples Without a WSDL Document

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

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

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

```
<?xml version="1.0" encoding="ASCII"?>
<composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200712"
           name="MyValueComposite">

  <service name="MyValueService">
    <interface.java interface="services.myvalue.MyValueService"/>
    <binding.ws/>
    ...
  </service>

  ...

  <reference name="StockQuoteService">
    <interface.java interface="services.stockquote.StockQuoteService"/>
    <binding.ws uri="http://www.example.org/StockQuoteService"/>
  </reference>
</composite>
```

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

```
<?xml version="1.0" encoding="ASCII"?>
<composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200712"
           name="MyValueComposite">

  <service name="MyValueService">
      <interface.java interface="services.myvalue.MyValueService"/>
    <binding.ws name="MyValueServiceSOAP11" requires="SOAP.1_1"/>
    <binding.ws requires="SOAP.1_2"/>
    ...
  </service>

  ...

  <reference name="StockQuoteService">
    <interface.java interface="services.stockquote.StockQuoteService"/>
    <binding.ws uri="http://www.example.org/StockQuoteService"
                requires="SOAP.1_2"/>
  </reference>
</composite>
```

3.3 Example PolicySet Providing The Conversation Intent

The following policy set applies to *binding.ws* and provides the conversation intent. The conversation intent is provided by using WS-ReliableMessaging [WS-RM] protocol which has a concept of a Sequence. This Sequence (which appears as a wsrm:Sequence SOAP header in the message) is used as a correlation mechanism, on the wire, to implement conversational semantics.

```
<policySet name="WSRM-Sequence-based-conversation"
        provides="sca:conversation"
        appliesTo="sca:binding.ws">
  <wsp:Policy>
    <wsrmp:RMAssertion
            xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200608"/>
  </wsp:Policy>
</policySet>
```

4 Transport Binding

The binding.ws element provides numerous ways to specify exactly how messages ought to be transmitted from or to the reference or service. Those ways include references to WSDL binding elements from the @wsdlElement attribute, policy intents, and even vendor extensions within the binding.ws element. However, all of those ways to indicate how messages get carried happen to be optional. This section describes the defaults to be used if the specific transport details are not otherwise specified.

4.1 Intents

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

- SOAP
  This indicates that messages MUST be transmitted using SOAP. One or more SOAP versions can be used.
- SOAP.1_1
  Messages MUST be transmitted using only SOAP 1.1.
- SOAP.1_2
  Messages MUST be transmitted using only SOAP 1.2.

4.2 Default Transport Binding Rules

4.2.1 WS-I Basic Profile Alignment

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

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

For a service element, the portType from the service's interface or derived from the service's interface MUST fit one of these two patterns. The rest of this section assumes the short-hand reference of an "rpc-literal" or "document-literal" pattern, depending on which of the two bullet points above it matches.

4.2.2 Default Transport Binding Rules

In the event that the transport details are not otherwise determined, an SCA runtime MUST enable the following configuration:

- HTTP-based transfer protocol
- Bindings for SOAP 1.1 MUST be provided and additional bindings MAY be provided, unless policy is applied that explicitly restricts this.
- "literal" format as described in section 3.5 of [WSDL11]
- For document literal pattern, each message uses "document" style, as per section 3.5 of [WSDL11].
- For rpc-literal pattern, each message uses "rpc" style, as per section 3.5 of [WSDL11]. In this case, the child elements of the SOAP Body element MUST be namespace qualified with a non-empty namespace name. This namespace SHOULD be the structural URI associated with the binding.
- For SOAP 1.1 messages, the SOAPAction HTTP header described in section 6.1.1 represents the empty string, in quotes ("").

- For SOAP 1.2 messages, the SOAP Action feature described in section 6.5 of [SOAP12Adjuncts] does not appear.
- All WSDL message parts are carried in the SOAP body

Formatted: Bullets and Numbering

5 SCA Web Services Callback Protocol

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

An SCA Runtime MAY implement SCA callback services over the Web services binding using WS-Addressing 1.0 capabilities, as described in this section.

To implement this protocol, an SCA binding follows the following rules.

Formatted: Bullets and Numbering

1. Every request message that invokes the forward interface MUST contain a Callback EPR. The Callback EPR MUST be carried in the request message in one of the following ways:
    a. If the request message contains the `wsa:From` SOAP header block then the `wsa:From` header block specifies the Callback EPR.
    b. If the `wsa:From` header block is not present then the `wsa:ReplyTo` header block specifies the Callback EPR.

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 WS-Addressing 1.0 SOAP Binding, Section 6.4.1 [WS-Addr-SOAP]. Such a fault can include additional `[Subsubcode] wsa:OnlyNonAnonymousAddressSupported`.

Formatted: Bullets and Numbering

2. A request message that invokes the forward interface can contain the `wsa:MessageID` SOAP header block. If there is a need to have the callback request message correlated to an individual forward request message, the `wsa:MessageID` SOAP header block can be used for this purpose.
3. When the service implementation invokes the callback interface, it MUST use the Callback EPR from a request message that invoked the forward interface, as specified in step 1. Once the Callback EPR is selected, the SCA runtime MUST follow the rules defined in WS-Addressing 1.0 Core Section 3.3 to invoke operations on the callback interface.

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. 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.

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.

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

5.1 Composability with WS-MakeConnection

Formatted: Bullets and Numbering

It is possible that the invoker of a service that uses a bidirectional interface has a binding that cannot accept connections for callbacks from a service (for example, when it has the `noListener` intent [SCA-Policy]). When this is the case, it is necessary for the binding to support a polling mechanism. An example of a polling mechanism is WS-MakeConnection. [WS-MC].

For the Web services binding, an SCA Runtime MAY implement SCA callback services over the Web services binding using WS-Addressing 1.0 capabilities combined with WS-MakeConnection, as described in this section. When an SCA runtime does implement such a capability, it MUST adhere to the rules described above (Section 5) and that of WS-MakeConnection, in addition to the rules described in this section.

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

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

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

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

5.2 Policy Assertion

Formatted: Bullets and Numbering

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

5.2.1 Assertion Model

Formatted: Bullets and Numbering

The WSCallback policy assertion indicates that the Web service client and the Web service MUST use SCA Web Services Callback Protocol to implement callbacks.

Specifically, the protocol determines the requirements on forward request message, the EPR used for callbacks and the requirements on the callback request message.

5.2.2 Normative Outline

The normative outline for the WSCallback assertion is:

```
<sca:WSCallback ...>
  ...
</sca:WSCallback>
```

The following describes the content model of the WSCallback element.

- **/sca:WSCallback**: A policy assertion that specifies that WSCallback protocol MUST be used when sending messages.

5.2.3 Assertion Attachment

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

- Endpoint Policy Subject

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

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:

- wsdl:portType

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

- wsdl:port
- wsdl:binding

5.2.4 Assertion Example

Table 2 lists an example use of the WSCallback policy assertion.

Table 2: Example policy with WSCallback policy assertion

```
(01)<wsdl:definitions
(02)     targetNamespace="example.com"
(03)     xmlns:tns="example.com"
(04)     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
(05)     xmlns:wsp="http://www.w3.org/ns/ws-policy"
(06)     xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200903"
(07)     xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
(08)
(09)  <wsp:UsingPolicy wsdl:required="true" />
(10)
(11)  <wsp:Policy wsu:Id="MyPolicy" >
(12)    <sca:WSCallback/>
```

Formatted: Bullets and Numbering

Formatted

Formatted: Bullets and Numbering

Formatted: Bullets and Numbering

Formatted: Bullets and Numbering

Formatted: Bullets and Numbering

```
(13) </wsp:Policy>
(14)
(15) <!-- omitted elements -->
(16)
(17) <wsdl:binding name="MyBinding" type="tns:MyPortType" >
(18)   <wsp:PolicyReference URI="#MyPolicy" />
(19)   <!-- omitted elements -->
(20) </wsdl:binding>
(21)
(22)</wsdl:definitions>
```

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

5.2.5 Security Considerations

Formatted: Bullets and Numbering

Policies and assertions SHOULD be signed to prevent tampering. 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. That is, a relying party shouldn't rely on a policy unless the policy is signed and presented with sufficient claims to pass the relying parties acceptance criteria.

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

~~5~~6 Conformance

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

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

a) SCA WS Binding XML Document

b) SCA Runtime

~~5.1~~6.1 SCA WS Binding XML Document

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

An SCA WS Binding XML document MUST be a conformant SCA Composite Document or a SCA ComponentType Document, as defined by the SCA Assembly specification [SCA-ASSEMBLY], and MUST comply with all the applicable requirements specified in this specification.

~~5.2~~6.2 SCA Runtime

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

1. The implementation MUST comply with all statements in Appendix XXX: Conformance Items related to an SCA Runtime, except for those that originate from Section 5, notably all "MUST" statements have to be implemented.

2. The implementation MAY support the Web Services Callback Protocol. If it does, it MUST comply with all statements in Appendix XXX: Conformance Items related to an SCA Runtime that originate from Section 5.

~~2.~~3. The implementation MUST conform to the SCA Assembly Model Specification Version 1.1 [SCA-Assembly], and to the SCA Policy Framework Version 1.1 [SCA-Policy].

~~3.~~4. The implementation MUST reject a SCA WS Binding XML Document that is not conformant per Section 6~~5~~.1.

Formatted: Bullets and Numbering

Formatted: Bullets and Numbering

Formatted: Bullets and Numbering

Formatted: Bullets and Numbering

Comment [ask3]: Page: 16
Moved to the top of the section and modified to match the Java CAA wordings

Comment [ask4]: Page: 16
Moved to section 5.1 and to section 2

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

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- (c) Copyright OASIS 2006, 2008 -->
<schema xmlns="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://docs.oasis-open.org/ns/opencsa/sca/200712"
   xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200712"
   xmlns:wsdli="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.xsd"/>

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

</schema>
```

Formatted: Bullets and Numbering

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

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- (c) Copyright OASIS 2006, 2008 -->
<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>
```

Formatted: Bullets and Numbering

~~B.~~C. Appendix - WSDL Generation

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

- component name = the value of the @name attribute of the component element containing the binding.ws element
- service name = the value of the @name attribute of the service element containing the binding.ws element
- binding name = the value of @name attribute of the binding.ws element, or the default if no @name attribute is present
- SOAP version = either "SOAP11" or "SOAP12" as appropriate

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

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

D. SCA Web Services Callback Protocol Message Examples

The message examples in this section are for a configuration that consists of a reference R that is wired to a Service S. S has a bidirectional interface and the binding used in both directions, forward and callback, is binding.ws configured for SOAP. Both the forward interface, as well as the callback interface, each, consists of only one one-way operation.

The following message exchanges take place between R and S:

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

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

R1:

```
<soap:Envelope ...>
 <soap:Header>
   <wsa:From>
     <wsa:Address>http://example.com/callback</wsa:Address>
     <wsa:ReferenceProperties>
       <myNS:SomeID>1</myNS:SomeID>
     </wsa:ReferenceProperties>
   </wsa:From>
   <wsa:MessageID>urn:uuid:f81d4fae-7dec-11d0-a765-
00a0c91e6bf6</wsa:messageID>
    ...
 </soap:Header>
 <soap:Body>
   ...
 </soap:Body>
</soap:Envelope>
```

S1, S2:

```
<soap:Envelope ...>
 <soap:Header>
    <wsa:To>http://example.com/callback</wsa:To>
    <myNS:SomeID>1</myNS:SomeID>
    <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback">urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6</wsa:RelatesTo>
     ...
 </soap:Header>
 <soap:Body>
    ...
 </soap:Body>
</soap:Envelope>
```

R2:

```
<soap:Envelope ...>
 <soap:Header>
    <wsa:From>
      <wsa:Address>http://example.com/callback</wsa:Address>
      <wsa:ReferenceProperties>
        <myNS:SomeID>1</myNS:SomeID>
      </wsa:ReferenceProperties>
    </wsa:From>
    <wsa:MessageID>urn:uuid:f81d4fae-8dec-11d0-a765-00a0c91e6bf6</wsa:messageID>
     ...
 </soap:Header>
 <soap:Body>
    ...
 </soap:Body>
</soap:Envelope>
```

S3:

```
<soap:Envelope ...>
 <soap:Header>
    <wsa:To>http://example.com/callback</wsa:To>
    <myNS:SomeID>1</myNS:SomeID>
    <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-bindings/ws/callback">
    urn:uuid:f81d4fae-8dec-11d0-a765-00a0c91e6bf6
    </wsa:RelatesTo>
     ...
 </soap:Header>
 <soap:Body>
    ...
 </soap:Body>
</soap:Envelope>
```

R3:

Formatted

Formatted

Formatted

```
<soap:Envelope ...>
 <soap:Header>
   <wsa:From>
     <wsa:Address>http://example.com/callback-other</wsa:Address>
     <wsa:ReferenceProperties>
       <myNS:SomeID>2</myNS:SomeID>
     </wsa:ReferenceProperties>
   </wsa:From>
   <wsa:MessageID>urn:uuid:f81d4fae-9dec-11d0-a765-
00a0c91e6bf6</wsa:messageID>
     ...
 </soap:Header>
 <soap:Body>
   ...
 </soap:Body>
</soap:Envelope>
```

S4, S5:

Formatted

```
<soap:Envelope ...>
 <soap:Header>
   <wsa:To>http://example.com/callback-other</wsa:To>
   <myNS:SomeID>2</myNS:SomeID>
   <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
bindings/ws/callback">urn:uuid:f81d4fae-9dec-11d0-a765-
00a0c91e6bf6</wsa:RelatesTo>
     ...
 </soap:Header>
 <soap:Body>
   ...
 </soap:Body>
</soap:Envelope>
```

D.1 Message Examples Using WS-MakeConnection

Formatted: Bullets and Numbering

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

R4:

Formatted

```
<soap:Envelope ...>
 <soap:Header>
   <wsa:From>
     <wsa:Address>http://docs.oasis-open.org/ws-
rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-446655440010</wsa:Address>
   </wsa:From>
   <wsa:MessageID>urn:uuid:f81d4fae-10dec-11d0-a765-
00a0c91e6bf6</wsa:messageID>
     ...
 </soap:Header>
 <soap:Body>
   ...
 </soap:Body>
</soap:Envelope>
```

MakeConnection polling message (from R to S):

```
<soap:Envelope ...>
 <soap:Header>
    <wsa:Action>http://docs.oasis-open.org/ws-
rx/wsmc/200702/MakeConnection</wsa:Action>
     ...
 </soap:Header>
 <soap:Body>
    <wsmc:MakeConnection>
       <wsmc:Address>http://docs.oasis-open.org/ws-
rx/wsmc/200702/anonymous?id=650e8400-f29b-11d4-a716-
446655440010</wsmc:Address>
    </wsmc:MakeConnection>
 </soap:Body>
</soap:Envelope>
```

S6:

```
<soap:Envelope ...>
 <soap:Header>
   <wsa:To>http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id=650e8400-
f29b-11d4-a716-446655440010</wsa:To>
   <wsa:RelatesTo RelationshipType="http://docs.oasis-open.org/opencsa/sca-
bindings/ws/callback">urn:uuid:f81d4fae-10dec-11d0-a765-
00a0c91e6bf6</wsa:RelatesTo>
    ...
 </soap:Header>
 <soap:Body>
   ...
 </soap:Body>
</soap:Envelope>
```

Formatted

Formatted

Formatted

~~C.~~E. Acknowledgements

Formatted: Bullets and Numbering

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

Participants:

[Participant Name, Affiliation | Individual Member]

[Participant Name, Affiliation | Individual Member]

~~D.~~F. Non-Normative Text

Formatted: Bullets and Numbering

~~E.~~G.Revision History

[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	* 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	* Completed application of resolution to issue 10 * Applied most of the editorial changes from Eric Johnson's review
4	2008-08-13	Anish Karmarkar	* 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	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