Service Component Architecture Web Service Binding Specification Version 1.1

Committee Draft 01 revision 21

2016th October, 2008

Specification URIs:
This Version:

http://docs.oasis-open.org/opencsa/sca-bindings/sca-binding-ws-1.1-spec-cd01-rev1.html

http://docs.oasis-open.org/opencsa/sca-bindings/sca-binding-ws-1.1-spec-cd01-rev1.doc

http://docs.oasis-open.org/opencsa/sca-bindings/sca-binding-ws-1.1-spec-cd01-rev1.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 supercedes:

- 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

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

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

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

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

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

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

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

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

Table of Contents

1 Introduction

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

The Web Service binding can point to an existing WSDL [WSDL] document, separately authored, that specifies the details of the WSDL binding and portType schema 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 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.

In most cases it is expected that a binding applied to a composite's reference will point to an existing WSDL document that describes the web service to be invoked. The binding applied to a composite's service ~~may~~can use either approach.

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.

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

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.html

[SCA-JCAA] http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec.html

[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/

[WS-Addr] http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/

2 Web Service Binding Schema

The normative web services binding XML Schema can be obtained by dereferencing the XML Schema namespace, and is also included for convenience in Appendix A. The <binding.ws> element MUST be valid according to its XML Schema.

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

```
<binding.ws name="xs:NCName"?
            requires="list of xs:QNames"?
            uri="xs:anyURI"?
            wsdlElement="xs:anyURI"?
            wsdli:wsdlLocation="list of xs:anyURI pairs"?
            ...>
   <wsa:EndpointReference>...</wsa: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/@uri*** - the resolution algorithm of Section 2.1 below describes how this attribute is interpreted.
- ***/binding.ws/@wsdlElement*** – ~~optional~~ non-required attribute that specifies the URI of a WSDL element. ~~The use of t~~This attribute ~~indicates that the SCA binding~~ 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, all the endpoints in the WSDL Service that have equivalent portTypes with the SCA service or reference ~~must~~ MUST be made available to the SCA service or reference by the SCA runtime.~~.~~
  - Port (WSDL 1.1):

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

    In this case, the ~~identified~~ port in the WSDL 1.1 Service identified by the <binding.ws> element ~~must~~ MUST have an equivalent portType with 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 ~~identified~~ endpoint in the WSDL 2.0 Service identified by the <binding.ws> element ~~must~~ 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 ~~identified~~ WSDL binding identified by the <binding.ws> element ~~must~~ MUST have an equivalent portType with 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 provided 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 is ~~optional~~not required. 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*** – ~~optional~~ non-required attribute that specifies the location(s) of the WSDL document(s) associated with specific namespace(s). This attribute ~~can~~ 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/wsa:EndpointReference*** – ~~optional~~ non-required 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.

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 *wsa:EndpointReference*,
2. The explicitly stated URI in the *@uri* attribute of the *binding.ws* element, which ~~may~~can be relative,
3. The implicit URI as defined by the Assembly specification

An SCA runtime MUST follow these rules to determine the URI at which an SCA service is hosted or an SCA reference is targetted.

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

To reference a WSDL document and also specify an EPR, the *@wsdlElement* attribute ~~must~~ MUST refer to a binding element in the WSDL ~~and not an endpoint or service~~.

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 interface for the service or reference is derived from the interface by the rules defined for that interface type.

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 ~~may~~can use appropriate standards, for example WS-I AP 1.0 or 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~~ SHOULD be provided by the SCA runtime. This ~~may~~ 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~~ 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 Web Service binding is specified using the wsdlElement attribute, the details of the binding are specified by the WSDL element referenced by the value of the attribute. WSDL elements allow for extensibility via elements as well as attribute. The Web Service binding allows the use of such extensibility in WSDL. Note that as a consequence of this, when using this form of Web Service binding, 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 ~~must be~~are listed as either *@alwaysProvides* or *@mayProvides* in the bindingType for *binding.ws*.

2.8 Intents and binding configuration

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 ~~must~~ 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 ~~may~~ 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 ~~must have~~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

This policy set applies to *binding.ws* and provides the conversation intent. The conversation intent is provided by using WS-ReliableMessaging 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 WSDL Generation

This section defines the rules that SHOULD be used by an SCA runtime for generation of a WSDL document that describes an SCA service with one or more web service bindings that require a SOAP binding.

A WSDL document ~~may~~ MAY be generated by an SCA runtime for an SCA service with non-SOAP web service bindings, or other bindings. For non-SOAP web service bindings that do not refer to an existing WSDL document, or non-web service bindings, the generation rules below ~~may~~can be considered a template, and a similar approach taken.

4.1 Intents

The following intents affect WSDL generation:

- soap
  This indicates that a SOAP binding is required. The SOAP binding ~~may~~ can be of any SOAP version, including multiple versions.
- soap.1_1
  A SOAP 1.1 binding only is required.
- soap.1_2
  A SOAP 1.2 binding only is required.

4.2 WSDL Service and Ports

A separate WSDL document is generated for each SCA service. Each has its own unique target namespace. This is to ensure that bindings on different services of the same component do not clash. The WSDL service has one or more ports for each web service binding on the SCA service that has a SOAP requirement, or that refers to an existing WSDL binding, depending on the requirements of the web service binding. Each of those ports has a single binding.

Additional ports and bindings ~~may~~ can be generated in this WSDL document for non-web service bindings, or web service bindings with non-SOAP requirements. The manner in which that is done is undefined.

The binding elements themselves ~~may~~can be generated as defined below, or ~~may~~ can be imported from existing WSDL documents in the case that the web service binding refers to the binding element of such a document.

The target namespace of the WSDL document, and of the service, ports and generated binding elements is:

Base System URI for HTTP / Component Name / Service Name

4.3 WSDL Bindings

The binding elements in the generated WSDL document are either defined within the document, derived from the requirements of the binding, or are imported from existing WSDL documents.

Generated bindings have the following fixed assumptions:

- use="literal" for input and output messages
- style="document" for the binding
- All faults map to soap:faults
- No header or headerFault elements are generated
- The transport is "http://schemas.xmlsoap.org/soap/http", unless the system provides intents for alternative transports
- The soap version is determined from the soap intents as defined above

4.3.1 SOAP versions

Where a web service binding requires a specific SOAP version, then a single WSDL port and SOAP binding of the appropriate version is generated.

Where no specific SOAP version is required, then one or more WSDL ports with associated SOAP bindings ~~may~~can be generated, depending on the level(s) supported in the target runtime.

4.4 WSDL portType

An SCA service has a single interface. This interface is always imported into the generated WSDL document. This ~~may~~can be done directly for WSDL-defined interfaces, or indirectly via a WSDL generated from the interface type for the service.

4.5 WSDL Generation Rules

The following is the formal definition of the generation of a WSDL document from an SCA service with one or more web service bindings, with either a SOAP requirement or existing WSDL document:

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="componentName.serviceName"
             targetNamespace="HTTP Base URI/componentName/serviceName"
             {(if any bindings require SOAP 1.1)
             xmlns:soap11="http://schemas.xmlsoap.org/wsdl/soap/"
             }
             {(if any bindings require SOAP 1.2)
             [xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"]
             }
             xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
             xmlns="http://schemas.xmlsoap.org/wsdl/">

  <import namespace="SCA service interface namespace"
          location="SCA service interface location"/>

  {(for each binding.ws element in the service with a WSDL, do the following:)
  <import namespace="existing WSDL binding namespace"
          location="existing WSDL binding location"/>
  }
```

```
  {(for each binding.ws element in the service without a WSDL, do the
following
    for each SOAP version required:)
  <binding name="/service/binding.ws[n]/@name+[.soapVersionPrefix]+'Binding'"
           type="SCA service interface portType name">
    <soapVersionPrefix:binding
transport="http://schemas.xmlsoap.org/soap/http"/>
    {(for each operation in the interface do the following:)
      <operation name="name-of-the-operation">
        <soapVersionPrefix:operation/>
        <input>
          <soapVersionPrefix:body use="literal"/>
        </input>
        {(if there is an output)
          <output>
            <soapVersionPrefix:body use="literal"/>
          </output>
        }
        {(if there is a fault)
          <fault>
            <soapVersionPrefix:fault name="name-of-the-fault"/>
          </fault>
        }
      </operation>
    }
  </binding>
  }

  <service name="/service/@name">
    {(for each binding.ws element in the service do the following for each
SOAP
      version required:)
      <port name="/service/binding.ws[n]/@name+[.soapVersionPrefix]+'Port'"

binding="/service/binding.ws[n]/@name+[.soapVersionPrefix]+'Binding'">
        <soapVersionPrefix:address location="/service/binding.ws[n]/@uri"/>
      </port>
    }
  </service>
</definitions>
```

5 Conformance

Any SCA runtime that claims to support this binding ~~must~~ MUST abide by the requirements of this specification.

The XML schema available at the namespace URI, defined by this specification, is considered to be authoritative and takes precedence over the XML Schema defined in the appendix of this document.

A. Web Services Binding Schema

```
<?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 ref="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>
```

B. Acknowledgements

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]

C. Non-Normative Text

D. 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.