



Service Component Architecture Web Service Binding Specification Version 1.1

Working Draft 03

12 June 2008

Specification URIs:

This Version:

<http://docs.oasis-open.org/sca-bindings/sca-wsbinding-1.1-spec-WD-03.html>
<http://docs.oasis-open.org/sca-bindings/sca-wsbinding-1.1-spec-WD-03.doc>
<http://docs.oasis-open.org/sca-bindings/sca-wsbinding-1.1-spec-WD-03.pdf> (Authoritative)

Previous Version:

Latest Version:

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

TBD

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

Copyright © OASIS® 2007. All Rights Reserved.

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

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

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

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

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

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

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

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

Table of Contents

1	Introduction	5
1.1	Terminology	5
1.2	Normative References	5
1.3	Non-Normative References	6
2	Web Service Binding Schema	7
2.1	Endpoint URI resolution	8
2.2	Interface mapping	8
2.3	Production of WSDL description for an SCA service	8
2.4	Additional binding configuration data	9
2.5	Web Service Binding and SOAP Intermediaries	9
2.6	Support for WSDL extensibility	9
2.7	Intents listed in the bindingType	9
3	Web Service Binding Examples	10
3.1	Example Using WSDL documents	10
3.2	Examples Without a WSDL Document	10
3.3	Example PolicySet Providing The Conversation Intent	12
4	WSDL Generation	13
4.1	Intents	13
4.2	WSDL Service and Ports	13
4.3	WSDL Bindings	13
4.3.1	SOAP versions	14
4.4	WSDL portType	14
4.5	WSDL Generation Rules	14
5	Conformance	16
A.	Web Services Binding Schema	17
B.	Acknowledgements	18
C.	Non-Normative Text	19
D.	Revision History	20

1 Introduction

The SCA Web Service binding specified in this document applies to the services and references of composites [1]. 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 [2] 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 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 [3] 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].

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.

[1] SCA Assembly Model Specification

<http://www.osoa.org/display/Main/Service+Component+Architecture+Specifications>

[2] WSDL Specification

WSDL 1.1: <http://www.w3.org/TR/wsd1>

WSDL 2.0: <http://www.w3.org/TR/wsd120/>

[3] WS-I 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>

41 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>
42 [4] JAX-WS Specification
43 <http://jcp.org/en/jsr/detail?id=224>
44 [5] SOAP specification
45 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>
46 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
47 [6] Web Services Addressing 1.0 – Core
48 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>
49 [7] Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language
50 <http://www.w3.org/TR/2007/REC-wsdl20-20070626>[8] SCA Java Common Annotations and API
51 Specification TBD
52
53 **TBD** **TBD**

54 **1.3 Non-Normative References**

55 **TB** **TBD**

2 Web Service Binding Schema

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

```
<binding.ws wsdlElement="xs:anyURI"?  
            wsdl:wsdlLocation="list of xs:anyURI pairs"?  
            ...>  
<wsa:EndpointReference>...</wsa:EndpointReference>*  
    ...  
</binding.ws>
```

- ***/binding.ws/@wsdlElement*** – optional attribute that specifies the URI of a WSDL element. The use of 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 be available to the SCA service or reference.
 - 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 must have an equivalent portType with the SCA service or reference.
 - 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 must have an equivalent portType with the SCA service or reference.
 - Binding:
`<WSDL-namespace-URI>#wsdl.binding(<binding-name>)`
In this case, the identified WSDL binding must have an equivalent portType with the SCA service or reference. In this case the endpoint address URI for the SCA service or reference must be provided via the URI attribute on the binding.
- ***/binding.ws/@wsdl:wsdlLocation*** – optional attribute that specifies the location(s) of the WSDL document(s) associated with specific namespace(s). This attribute can be specified 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 [7].
- ***/binding.ws/wsa:EndpointReference*** – optional WS-Addressing [6] 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

98 wsdlElement attribute value MUST be of the 'Binding' form as specified above, i.e.
99 <WSDL-namespace-URI>#wsdl.binding(<binding-name>).

- 100 • **/binding.ws/@{any}** - this is an extensibility mechanism to allow extensibility via
101 attributes.
- 102 • **/binding.ws/any** – this is an extensibility mechanism to allow extensibility via
103 elements.

104 2.1 Endpoint URI resolution

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

- 107 1. The URIs in the endpoint(s) of the referenced WSDL
108 or
109 The URI specified by the wsa:Address element of the wsa:EndpointReference,
- 110 2. The explicitly stated URI in the "uri" attribute of the binding.ws element, which
111 may be relative,
- 112 3. The implicit URI as defined by the Assembly specification

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

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

120 2.2 Interface mapping

121 When binding.ws is used on a service or reference with an interface that is not defined
122 by interface.wsdl, then a WSDL interface for the service or reference is derived from the
123 interface by the rules defined for that interface type.

124 For example, for interface.java, the mapping to a WSDL portType is as defined in the
125 SCA Java Common Annotations and API Specification [8].

126 Binding.ws implementations may use appropriate standards, for example WS-I AP 1.0 or
127 MTOM, to map interface parameters to binary attachments transparently to the target
128 component.

129

130 2.3 Production of WSDL description for an SCA service

131 Any service with one or more web service bindings with HTTP endpoints SHOULD return
132 a WSDL description of the service in response to an HTTP GET request with the "?wsdl"
133 suffix to that HTTP endpoint. If none of the web service bindings have HTTP endpoints,
134 then some other means of obtaining the WSDL description of the service should be
135 provided. This may include out of band mechanisms, for example publication to a UDDI
136 registry.

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

140

141 **2.4 Additional binding configuration data**

142 SCA runtime implementations may provide additional metadata that is associated with a
143 web service binding, for example to enable JAX-WS [4] handlers to be executed as part
144 of the target component dispatch. The specification of such metadata is SCA runtime-
145 specific and is outside of the scope of this document.

146

147 **2.5 Web Service Binding and SOAP Intermediaries**

148 The Web Service binding does not provide any direct or explicit support for SOAP
149 intermediaries [5].

150

151 **2.6 Support for WSDL extensibility**

152 When a Web Service binding is specified using the wsdlElement attribute, the details of
153 the binding are specified by the WSDL element referenced by the value of the attribute.
154 WSDL elements allow for extensibility via elements as well as attribute. The Web Service
155 binding allows the use of such extensibility in WSDL. Note that as a consequence of this,
156 when using this form of Web Service binding, it is not possible to determine whether the
157 binding is supported by the SCA runtime without parsing the referenced WSDL element
158 and its dependent elements.

159 **2.7 Intents listed in the bindingType**

160 This specification places no requirements on the intents that must be listed as either @alwaysProvides or
161 @mayProvides in the bindingType for binding.ws.

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 conform 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 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://www.osoa.org/xmlns/sca/1.0" name="MyValueComposite">
  <service name="MyValueService">
    <interface.java interface="services.myvalue.MyValueService"/>
    <binding.ws wsdlElement="http://www.myvalue.org/MyValueService#
wsdl.endpoint(MyValueService/MyValueServiceSOAP)"/>
    ...
  </service>
  ...
  <reference name="StockQuoteReference1">
    <interface.java interface="services.stockquote.StockQuoteService"/>
    <binding.ws wsdlElement="http://www.stockquote.org/StockQuoteService#
wsdl.service(StockQuoteService)"
wsdli:wsdlLocation="http://www.stockquote.org/StockQuoteService
http://www.stockquote.org/StockQuoteService.wsdl"/>
  </reference>
  <reference name="StockQuoteReference2">
    <interface.java interface="services.stockquote.StockQuoteService"/>
    <binding.ws wsdlElement="http://www.stockquote.org/StockQuoteService#
wsdl.binding(StockQuoteBinding)"
wsdli:wsdlLocation="http://www.stockquote.org/StockQuoteService
http://www.stockquote.org/StockQuoteService.wsdl"
uri="http://www.stockquote.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

211 service and reference each have a single binding element, whose name defaults to the
212 service/reference name.

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

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

221

```
222 <?xml version="1.0" encoding="ASCII"?>
223 <composite xmlns="http://www.oesa.org/xmlns/sca/1.0"
224           name="MyValueComposite">
225
226   <service name="MyValueService">
227     <interface.java interface="services.myvalue.MyValueService"/>
228     <binding.ws/>
229     ...
230   </service>
231
232   ...
233
234   <reference name="StockQuoteService">
235     <interface.java interface="services.stockquote.StockQuoteService"/>
236     <binding.ws uri="http://www.sqs.com/StockQuoteService"/>
237   </reference>
238 </composite>
```

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

244

```
245
246 <?xml version="1.0" encoding="ASCII"?>
247 <composite xmlns="http://www.oesa.org/xmlns/sca/1.0"
248           name="MyValueComposite">
249
250   <service name="MyValueService">
251     <interface.java interface="services.myvalue.MyValueService"/>
252     <binding.ws name="MyValueServiceSOAP11" requires="soap.1_1"/>
253     <binding.ws requires="soap.1_2"/>
254     ...
255   </service>
256
257   ...
258
259   <reference name="StockQuoteService">
260     <interface.java interface="services.stockquote.StockQuoteService"/>
261     <binding.ws uri="http://www.sqs.com/StockQuoteService"
262                requires="soap.1_2"/>
263   </reference>
264 </composite>
```

265

266 3.3 Example PolicySet Providing The Conversation Intent

267 This policy set applies to binding.ws and provides the conversation intent. The
268 conversation intent is provided by using WS-ReliableMessaging protocol which has a
269 concept of a Sequence. This Sequence (which appears as a wsrmp:Sequence SOAP
270 header in the message) is used as a correlation mechanism, on the wire, to implement
271 conversational semantics.

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

280

281 4 WSDL Generation

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

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

289

290 4.1 Intents

291 The following intents affect WSDL generation:

- 292 • soap
293 This indicates that a SOAP binding is required. The SOAP binding may be of any
294 SOAP version, including multiple versions.
- 295 • soap.1_1
296 A SOAP 1.1 binding only is required.
- 297 • soap.1_2
298 A SOAP 1.2 binding only is required.

299

300 4.2 WSDL Service and Ports

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

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

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

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

315 Base System URI for HTTP / Component Name / Service Name

316

317 4.3 WSDL Bindings

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

- 321 Generated bindings have the following fixed assumptions:
- 322 • use="literal" for input and output messages
 - 323 • style="document" for the binding
 - 324 • All faults map to soap:faults
 - 325 • No header or headerFault elements are generated
 - 326 • The transport is "http://schemas.xmlsoap.org/soap/http", unless the system
 - 327 provides intents for alternative transports
 - 328 • The soap version is determined from the soap intents as defined above

329

330 4.3.1 SOAP versions

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

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

336

337 4.4 WSDL portType

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

341

342 4.5 WSDL Generation Rules

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

```
346 <?xml version="1.0" encoding="UTF-8"?>
347 <definitions name="componentName.serviceName"
348             targetNamespace="HTTP Base URI/componentName/serviceName"
349             {(if any bindings require SOAP 1.1)
350             xmlns:soap11="http://schemas.xmlsoap.org/wsdl/soap/"
351             }
352             {(if any bindings require SOAP 1.2)
353             [xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"]
354             }
355             xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
356             xmlns="http://schemas.xmlsoap.org/wsdl/">
357
358     <import namespace="SCA service interface namespace"
359           location="SCA service interface location"/>
360
361     {(for each binding.ws element in the service with a WSDL, do the following:)
362     <import namespace="existing WSDL binding namespace"
363           location="existing WSDL binding location"/>
364     }
365
```

```

366      {(for each binding.ws element in the service without a WSDL, do the
367 following
368     for each SOAP version required:)
369     <binding name="/service/binding.ws[n]/@name+[.soapVersionPrefix]+'Binding'"
370         type="SCA service interface portType name">
371         <soapVersionPrefix:binding
372 transport="http://schemas.xmlsoap.org/soap/http"/>
373         {(for each operation in the interface do the following:)
374         <operation name="name-of-the-operation">
375         <soapVersionPrefix:operation/>
376         <input>
377         <soapVersionPrefix:body use="literal"/>
378         </input>
379         {(if there is an output)
380         <output>
381         <soapVersionPrefix:body use="literal"/>
382         </output>
383         }
384         {(if there is a fault)
385         <fault>
386         <soapVersionPrefix:fault name="name-of-the-fault"/>
387         </fault>
388         }
389         </operation>
390         }
391     </binding>
392     }
393
394     <service name="/service/@name">
395     {(for each binding.ws element in the service do the following for each
396 SOAP
397     version required:)
398     <port name="/service/binding.ws[n]/@name+[.soapVersionPrefix]+'Port'"
399
400 binding="/service/binding.ws[n]/@name+[.soapVersionPrefix]+'Binding'">
401     <soapVersionPrefix:address location="/service/binding.ws[n]/@uri"/>
402     </port>
403     }
404     </service>
405 </definitions>

```

406 **5 Conformance**

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

408 TBD

A. Web Services Binding Schema

```
410 <?xml version="1.0" encoding="UTF-8"?>
411 <!-- (c) Copyright OASIS 2007 -->
412 <schema xmlns="http://www.w3.org/2001/XMLSchema"
413         targetNamespace="[TO BE CONFIRMED]"
414         xmlns:sca="[TO BE CONFIRMED]"
415         xmlns:wsdli="http://www.w3.org/ns/wsdli-instance"
416         xmlns:wsa="http://www.w3.org/2005/08/addressing"
417         elementFormDefault="qualified">
418
419     <import namespace="http://www.w3.org/ns/wsdli-instance"
420             schemaLocation="http://www.w3.org/2007/05/wsdli/wsdli20-instance.xsd"
421             />
422     <import namespace="http://www.w3.org/2005/08/addressing"
423             schemaLocation="http://www.w3.org/2005/08/addressing/ws-addr.xsd"
424             />
425     <include schemaLocation="sca-core.xsd"/>
426
427     <element name="binding.ws" type="sca:WebServiceBinding"
428             substitutionGroup="sca:binding"/>
429     <complexType name="WebServiceBinding">
430         <complexContent>
431             <extension base="sca:Binding">
432                 <sequence>
433                     <element ref="wsa:EndpointReference" minOccurs="0"
434                             maxOccurs="unbounded"/>
435                     <any namespace="##other" processContents="lax"
436                         minOccurs="0"
437                             maxOccurs="unbounded"/>
438                 </sequence>
439                 <attribute name="wsdlElement" type="anyURI" use="optional"/>
440                 <attribute ref="wsdli:wSDLLocation" use="optional"/>
441                 <anyAttribute namespace="##any" processContents="lax"/>
442             </extension>
443         </complexContent>
444     </complexType>
445 </schema>
```

446

447 **B. Acknowledgements**

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

450 **Participants:**

451 [Participant Name, Affiliation | Individual Member]

452 [Participant Name, Affiliation | Individual Member]

453

455

D. Revision History

456

[optional; should not be included in OASIS Standards]

457

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

458

459