

SCA Service Component Architecture

Web Service Binding Specification

SCA Version 1.00, March 21 2007

Technical Contacts:

| | |
|------------------|---------------------|
| Simon Holdsworth | IBM Corporation |
| Sabin Ielceanu | TIBCO Software Inc. |
| Anish Karmarkar | Oracle |
| Mark Little | Red Hat |
| Sanjay Patil | SAP AG |
| Michael Rowley | BEA |

Copyright Notice

© Copyright BEA Systems, Inc., Cape Clear Software, International Business Machines Corp, Interface21, IONA Technologies, Oracle, Primeton Technologies, Progress Software, Red Hat, Rogue Wave Software, SAP AG., Siemens AG., Software AG., Sun Microsystems, Inc., Sybase Inc., TIBCO Software Inc., 2005, 2007. All rights reserved.

License

The Service Component Architecture Specification is being provided by the copyright holders under the following license. By using and/or copying this work, you agree that you have read, understood and will comply with the following terms and conditions:

Permission to copy, display and distribute the Service Component Architecture Specification and/or portions thereof, without modification, in any medium without fee or royalty is hereby granted, provided that you include the following on ALL copies of the Service Component Architecture Specification, or portions thereof, that you make:

1. A link or URL to the Service Component Architecture Specification at this location:
 - <http://www.osoa.org/display/Main/Service+Component+Architecture+Specifications>
2. The full text of the copyright notice as shown in the Service Component Architecture Specification.

BEA, Cape Clear, IBM, Interface21, IONA, Oracle, Primeton, Progress Software, Red Hat, Rogue Wave, SAP, SIEMENS AG, Software AG., Sun Microsystems, Sybase, TIBCO (collectively, the "Authors") agree to grant you a royalty-free license, under reasonable, non-discriminatory terms and conditions to patents that they deem necessary to implement the Service Component Architecture Specification.

THE Service Component Architecture SPECIFICATION IS PROVIDED "AS IS," AND THE AUTHORS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, REGARDING THIS SPECIFICATION AND THE IMPLEMENTATION OF ITS CONTENTS, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT OR TITLE.

THE AUTHORS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THE Service Components Architecture SPECIFICATION.

The name and trademarks of the Authors may NOT be used in any manner, including advertising or publicity pertaining to the Service Component Architecture Specification or its contents without specific, written prior permission. Title to copyright in the Service Component Architecture Specification will at all times remain with the Authors.

No other rights are granted by implication, estoppel or otherwise.

Status of this Document

This specification may change before final release and you are cautioned against relying on the content of this specification. The authors are currently soliciting your contributions and suggestions. Licenses are available for the purposes of feedback and (optionally) for implementation.

IBM is a registered trademark of International Business Machines Corporation in the United States, other countries, or both.

BEA is a registered trademark of BEA Systems, Inc.

Cape Clear is a registered trademark of Cape Clear Software

IONA and IONA Technologies are registered trademarks of IONA Technologies plc.

Oracle is a registered trademark of Oracle USA, Inc.

Progress is a registered trademark of Progress Software Corporation

Primeton is a registered trademark of Primeton Technologies, Ltd.

Red Hat is a registered trademark of Red Hat Inc.

Rogue Wave is a registered trademark of Quovadx, Inc

SAP is a registered trademark of SAP AG.

SIEMENS is a registered trademark of SIEMENS AG

Software AG is a registered trademark of Software AG

Sun and Sun Microsystems are registered trademarks of Sun Microsystems, Inc.

Sybase is a registered trademark of Sybase, Inc.

TIBCO is a registered trademark of TIBCO Software Inc.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Table of Contents

| | |
|---|-----|
| Copyright Notice | ii |
| License | ii |
| Status of this Document | iii |
| Table of Contents | iv |
| 1 Overview | 1 |
| 1.1 Introduction | 1 |
| 2 Web Service Binding | 2 |
| 2.1 Web Service Binding Schema | 2 |
| 2.1.1 Endpoint URI resolution | 3 |
| 2.1.2 Interface mapping | 3 |
| 2.1.3 Production of WSDL description for an SCA service | 3 |
| 2.1.4 Additional binding configuration data | 3 |
| 2.1.5 Web Service Binding and SOAP Intermediaries | 4 |
| 2.1.6 Support for WSDL extensibility | 4 |
| 2.2 Web Service Binding Examples | 4 |
| 2.2.1 Example Using WSDL documents | 4 |
| 2.2.2 Examples Without a WSDL Document | 5 |
| 2.2.3 Example PolicySet Providing The Conversation Intent | 6 |
| 2.3 WSDL Generation | 6 |
| 2.3.1 Intents | 6 |
| 2.3.2 WSDL Service and Ports | 6 |
| 2.3.3 WSDL Bindings | 7 |
| 2.3.3.1 SOAP versions | 7 |
| 2.3.4 WSDL PortType | 7 |
| 2.3.5 WSDL Generation Rules | 7 |
| 3 Web Services Binding Schema | 9 |
| 4 References | 10 |

1 Overview

1.1 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.

24 2 Web Service Binding

25

26 2.1 Web Service Binding Schema

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

```
28 <binding.ws wsdlElement="xs:anyURI"?
29         wsdlLocation="list of xs:anyURI"?
30         ...>
31     <wsa:EndpointReference>...</wsa:EndpointReference>*
32     ...
33 </binding.ws>
```

34

- 35 • ***/binding.ws/@wsdlElement*** – optional attribute that specifies the URI of a WSDL element.
36 The use of this attribute indicates that the SCA binding points to the specified element in an
37 existing WSDL document. The URI can have the following forms:

- 38 ○ Service:

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

40 In this case, all the endpoints in the WSDL Service that have equivalent PortTypes with
41 the SCA service or reference must be available to the SCA service or reference.

- 42 ○ Port (WSDL 1.1):

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

44 In this case, the identified port in the WSDL 1.1 Service must have an equivalent
45 PortType with the SCA service or reference.

- 46 ○ Endpoint (WSDL 2.0):

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

48 In this case, the identified endpoint in the WSDL 2.0 Service must have an equivalent
49 PortType with the SCA service or reference.

- 50 ○ Binding:

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

52 In this case, the identified WSDL binding must have an equivalent PortType with the SCA
53 service or reference. In this case the endpoint address URI for the SCA service or
54 reference must be provided via the URI attribute on the binding.

- 55 • ***/binding.ws/@wsdl:wsdlLocation*** – optional attribute that specifies the location of the
56 WSDL document. This attribute can be specified in the event that the `<WSDL-namespace-
57 URI>` in the 'endpoint' attribute is not dereferencable, or when the intended WSDL document
58 is to be found at a different location than the one pointed to by the `<WSDL-namespace-
59 URI>`. The use of this attribute indicates that the WSDL binding points to an existing WSDL
60 document.

- 61 • ***/binding.ws/wsa:EndpointReference*** – optional WS-Addressing [6] EndpointReference
62 that specifies the endpoint for the service or reference. When this element is present along
63 with the `wsdlElement` attribute on the parent element, the `wsdlElement` attribute value MUST
64 be of the 'Binding' form as specified above, i.e. `<WSDL-namespace-
65 URI>#wsdl.binding(<binding-name>)`.

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

- */binding.ws/any* – this is an extensibility mechanism to allow extensibility via elements.

2.1.1 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 be relative,
3. The implicit URI as defined by the Assembly specification

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 can 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 refer to a binding element in the WSDL and not an endpoint or service.

2.1.2 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 Assembly Specification [1].

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

2.1.3 Production of WSDL description for an SCA service

Any service 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. This may include out of band mechanisms, for example publication to a UDDI registry.

Refer to section 2.3 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.

2.1.4 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 [4] 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.

111 2.1.5 Web Service Binding and SOAP Intermediaries

112 The Web Service binding does not provide any direct or explicit support for SOAP intermediaries
113 [5].

114

115 2.1.6 Support for WSDL extensibility

116 When a Web Service binding is specified using the `wsdlElement` attribute, the details of the
117 binding are specified by the WSDL element referenced by the value of the attribute. WSDL
118 elements allow for extensibility via elements as well as attribute. The Web Service binding does
119 not curtail the use of such extensibility in WSDL. Note that as a consequence of this, when using
120 this form of Web Service binding, it is not possible to determine whether the binding is supported
121 by the SCA runtime without parsing the referenced WSDL element and its dependent elements.

122

123 2.2 Web Service Binding Examples

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

127

128 2.2.1 Example Using WSDL documents

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

132 The service's binding is defined by the WSDL document associated with the given URI. This
133 service must be conformant with the WS-I basic profile 1.1.

134 The reference's first binding is defined by the specified WSDL service in the WSDL document at
135 the given location. The reference may use any of the WSDL service's ports/endpoints to invoke
136 the target service. The reference's second binding is defined by the specified WSDL binding. The
137 specific endpoint URI to be invoked is provided via the `URI` attribute.

138

```
139 <?xml version="1.0" encoding="ASCII"?>
140 <composite xmlns="http://www.osoa.org/xmlns/sca/1.0" name="MyValueComposite">
141   <service name="MyValueService">
142     <interface.java interface="services.myvalue.MyValueService"/>
143     <binding.ws wsdlElement="http://www.myvalue.org/MyValueService#
144                   wsdl.endpoint(MyValueService/MyValueServiceSOAP)"/>
145     ...
146   </service>
147
148   ...
149
150   <reference name="StockQuoteReference1">
151     <interface.java interface="services.stockquote.StockQuoteService"/>
152     <binding.ws wsdlElement="http://www.stockquote.org/StockQuoteService#
153                   wsdl.service(StockQuoteService)"
154                   wsdl:wsdlLocation="http://www.stockquote.org/StockQuoteService
155                   http://www.stockquote.org/StockQuoteService.wsdl"/>
156   </reference>
157
158   <reference name="StockQuoteReference2">
159     <interface.java interface="services.stockquote.StockQuoteService"/>
160     <binding.ws wsdlElement="http://www.stockquote.org/StockQuoteService#
```



```

161         wsdl.binding(StockQuoteBinding) "
162
163         wsdl:wsdlLocation="http://www.stockquote.org/StockQuoteService
164             http://www.stockquote.org/StockQuoteService.wsdl"/>
165         uri="http://www.stockquote.org/StockQuoteService5"/>
166     </reference>
167 </composite>
168

```

2.2.2 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 BP 1.1 compliant binding, and using the default options for mapping the Java interface to a WSDL port type.

The reference indicates a service to be invoked which must have 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.

```

182 <?xml version="1.0" encoding="ASCII"?>
183 <composite xmlns="http://www.oesa.org/xmlns/sca/1.0"
184     name="MyValueComposite">
185     <service name="MyValueService">
186         <interface.java interface="services.myvalue.MyValueService"/>
187         <binding.ws/>
188         ...
189     </service>
190     ...
191     <reference name="StockQuoteService">
192         <interface.java interface="services.stockquote.StockQuoteService"/>
193         <binding.ws uri="http://www.sqs.com/StockQuoteService"/>
194     </reference>
195 </composite>
196

```

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.

```

206 <?xml version="1.0" encoding="ASCII"?>
207 <composite xmlns="http://www.oesa.org/xmlns/sca/1.0"
208     name="MyValueComposite">
209     <service name="MyValueService">
210         <interface.java interface="services.myvalue.MyValueService"/>
211         <binding.ws name="MyValueServiceSOAP11" requires="soap/1.1"/>
212         <binding.ws requires="soap/1.2"/>
213     </service>
214     <reference name="StockQuoteService">
215         <binding.ws requires="soap/1.2"/>
216     </reference>
217 </composite>
218

```

```

215     ...
216 </service>
217
218     ...
219
220 <reference name="StockQuoteService">
221     <interface.java interface="services.stockquote.StockQuoteService"/>
222     <binding.ws uri="http://www.sqs.com/StockQuoteService"
223               requires="soap/1.2"/>
224 </reference>
225 </composite>
226

```

2.2.3 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 wsrmp:Sequence SOAP header in the message) is used as a correlation mechanism, on the wire, to implement conversational semantics.

```

232 <policySet name="WSRM-Sequence-based-conversation"
233           provides="sca:conversation"
234           appliesTo="sca:binding.ws">
235   <wsp:Policy>
236     <wsrmp:RMAssertion
237       xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200608"/>
238   </wsp:Policy>
239 </policySet>
240

```

2.3 WSDL Generation

This section defines the rules that SHOULD be used 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 be generated 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 be considered a template, and a similar approach taken.

2.3.1 Intents

The following intents affect WSDL generation:

- 251 • soap
252 This indicates that a SOAP binding is required. The SOAP binding may be of any SOAP
253 version, including multiple versions.
- 254 • soap.1_1
255 A SOAP 1.1 binding only is required.
- 256 • soap.1_2
257 A SOAP 1.2 binding only is required.

2.3.2 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

263 that has a SOAP requirement, or that refers to an existing WSDL binding, depending on the
264 requirements of the web service binding. Each of those ports has a single binding.

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

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

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

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

275 2.3.3 WSDL Bindings

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

278 Generated bindings have the following fixed assumptions:

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

287 2.3.3.1 SOAP versions

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

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

293 2.3.4 WSDL PortType

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

298 2.3.5 WSDL Generation Rules

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

```
302 <?xml version="1.0" encoding="UTF-8"?>
303 <definitions name="componentName/serviceName"
304             targetNamespace="HTTP Base URI/componentName/serviceName"
305             {(if any bindings require SOAP 1.1)
```

```

306         xmlns:soap11="http://schemas.xmlsoap.org/wsdl/soap/"
307     }
308     {(if any bindings require SOAP 1.2)
309     [xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"]
310     }
311     xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
312     xmlns="http://schemas.xmlsoap.org/wsdl/">
313
314     <import namespace="SCA service interface namespace"
315         location="SCA service interface location"/>
316
317     {(for each binding.ws element in the service with a WSDL, do the following:)
318     <import namespace="existing WSDL binding namespace"
319         location="existing WSDL binding location"/>
320     }
321
322     {(for each binding.ws element in the service without a WSDL, do the following
323     for each SOAP version required:)
324     <binding name="/service/binding.ws[n]/@name+[/soapVersionPrefix]+'Binding'"
325         type="SCA service interface portType name">
326         <soapVersionPrefix:binding transport="http://schemas.xmlsoap.org/soap/http"/>
327         {(for each operation in the interface do the following:)
328         <operation name="name-of-the-operation">
329             <soapVersionPrefix:operation/>
330             <input>
331                 <soapVersionPrefix:body use="literal"/>
332             </input>
333             {(if there is an output)
334             <output>
335                 <soapVersionPrefix:body use="literal"/>
336             </output>
337             }
338             {(if there is a fault)
339             <fault>
340                 <soapVersionPrefix:fault name="name-of-the-fault"/>
341             </fault>
342             }
343         </operation>
344     }
345     </binding>
346 }
347
348 <service name="/service/@name">
349     {(for each binding.ws element in the service do the following for each SOAP
350     version required:)
351     <port name="/service/binding.ws[n]/@name+[/soapVersionPrefix]+'Port'"
352         binding="/service/binding.ws[n]/@name+[/soapVersionPrefix]+'Binding'">
353         <soapVersionPrefix:address location="/service/binding.ws[n]/@uri"/>
354     </port>
355     }
356 </service>
357 </definitions>

```

3 Web Services Binding Schema

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- (c) Copyright SCA Collaboration 2006 -->
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.oesa.org/xmlns/sca/1.0"
  xmlns:sca="http://www.oesa.org/xmlns/sca/1.0"
  xmlns:w3="http://www.w3.org/2004/08/wsdli-instance"
  xmlns:wsa="http://www.w3.org/2004/12/addressing"
  elementFormDefault="qualified">

  <import namespace="http://www.w3.org/2004/08/wsdli-instance"
    schemaLocation="wsdli.xsd" />
  <import namespace="http://www.w3.org/2004/12/addressing"
    schemaLocation="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="w3:wsdlLocation" use="optional"/>
        <anyAttribute namespace="##any" processContents="lax"/>
      </extension>
    </complexContent>
  </complexType>
</schema>
```

4 References

394
395
396 [1] SCA Assembly Model Specification

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

398
399 [2] WSDL Specification

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

401 WSDL 2.0: <http://www.w3.org/TR/wsdl20/>

402
403 [3] WS-I Profiles

404 <http://www.ws-i.org/Profiles/BasicProfile-1.1.html>

405 <http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html>

406 <http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html>

407 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>

408
409 [4] JAX-WS Specification

410 <http://jcp.org/en/jsr/detail?id=224>

411
412 [5] SOAP specification

413 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

414 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

415
416 [6] Web Services Addressing 1.0 – Core

417 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>

418