



Test Suite Adaptation for SCA Assembly Model Version 1.1 Specification

Committee Draft 01 / Public Review 01

20 July 2010

Specification URIs:

This Version:

<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testsuite-adaptation-cd01.html>

<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testsuite-adaptation-cd01.odt>

<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testsuite-adaptation-cd01.pdf> (Authoritative)

Previous Version:

N/A

Latest Version:

<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testsuite-adaptation.html>

<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testsuite-adaptation.odt>

<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testsuite-adaptation.pdf>
(Authoritative)

Technical Committee:

OASIS Service Component Architecture / Assembly (SCA-Assembly) TC

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=sca-assembly

Chair(s):

Martin Chapman, Oracle

Mike Edwards, IBM

Editor(s):

Bryan Aupperle

Dave Booz

Mike Edwards

Jeff Estefan

Related Work:

This document is related to:

- Service Component Architecture Assembly Specification Version 1.1
<http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-cd05.pdf>

Declared XML Namespace(s):

none

Abstract:

This document defines the requirements for adaptation of the SCA Assembly Test Suite to use a new SCA implementation type that is provided by a conforming SCA Runtime. The SCA Runtime needs to pass the SCA Assembly Test Suite without failures. Where the SCA Runtime supports an implementation type that is not currently supported by the SCA Assembly Test Suite, it is necessary for the test suite to be adapted to use that implementation type, before the test suite can be run successfully against that SCA Runtime.

Status:

This document was last revised or approved by the OASIS Service Component Architecture / Assembly (SCA-Assembly) 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-assembly/>.

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-assembly/ipr.php>).

The non-normative errata page for this specification is located at <http://www.oasis-open.org/committees/sca-assembly/>

Notices

Copyright © OASIS® 2010. 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", "Service Component Architecture" 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 SCA Assembly Test Suite Organization.....	7
2.1 Implementation Type Dependent Artifacts.....	8
2.1.1 Implementation-Dependent Composite Files.....	10
2.1.2 Implementations.....	13
2.1.3 Interfaces.....	13
2.1.4 sca-contribution.xml file.....	13
2.2 The Test Client.....	13
2.2.1 Configuring the Test Client.....	14
2.2.2 Configuration of Each TestCase.....	15
2.2.3 Creating an Alternative Test Client.....	16
3 Conformance.....	17
A.Required Artifacts.....	18
A.1.Composite Files in the Implementation Type Contribution.....	18
B.Conformance Items.....	25
B.1.Mandatory Items.....	25
C.Acknowledgments.....	26
D.Revision History.....	27

1 Introduction

[All text is normative unless otherwise indicated.]

This document defines the requirements for adaptation of the SCA Assembly Test Suite to use a new SCA implementation type that is provided by a conforming SCA Runtime. The SCA Runtime needs to pass the SCA Assembly Test Suite without failures. Where the SCA Runtime supports an implementation type that is not currently supported by the SCA Assembly Test Suite, it is necessary for the test suite to be adapted to use that implementation type, before the test suite can be run successfully against that SCA Runtime.

The SCA Assembly Test Suite is designed for adaptation to new implementation types. The test suite is divided into two groups of artifacts, handled by means of separate SCA contributions:

- Implementation type-independent artifacts, largely consisting of SCA composites, with associated supporting artifacts such as interfaces provided as WSDL declarations.
- Implementation type-dependent artifacts, which consist of implementations in the relevant implementation type, plus SCA composites which directly wrap those implementations, and other associated artifacts which can include interfaces provided in a language which is natural for the implementation type (eg. Java implementation classes have their interfaces declared as Java interfaces)

The SCA Assembly Test Suite is described through two documents:

- The TestCases document [SCA-TestCases], which describes the detail of the testcases themselves and their related artifacts.
- The Test Assertions document [SCA-Assertions], which describes a set of assertions which must be validated by the test suite. The test assertions are derived directly from the normative statements of the SCA Assembly specification. The test assertions effectively render the requirements of the specification as a series of assertions that can be cast into the form of one or more testcases which then validate conformance with the specification.

This document largely concerns itself with the TestCases document and the associated sets of artifacts that make up the test suite itself. The Test Assertions document can be used as a guide to the intention of each testcase and is the way in which each testcase can be linked back to appropriate normative statements in the SCA Assembly specification [SCA-Assembly].

1.1 Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119 [RFC 2119].

1.2 Normative References

- | | |
|------------------------|---|
| [RFC 2119] | S. Bradner. <i>Key words for use in RFCs to Indicate Requirement Levels</i> . IETF RFC 2119, March 1997. http://www.ietf.org/rfc/rfc2119.txt . |
| [SCA-Assembly] | OASIS Committee Draft 05, Service Component Architecture Assembly Model Specification Version 1.1, January 2010.
http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-cd05.pdf |
| [SCA-TestCases] | OASIS Committee Draft 01, TestCases for the SCA Assembly Model V1.1 Specification, June 2009. |

43 [http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testcases-](http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testcases-cd01.pdf)
44 [cd01.pdf](http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-testcases-cd01.pdf)
45 **[SCA-Assertions]** OASIS Committee Draft 02, Test Assertions for the SCA Assembly Model
46 Version 1.1 Specification, June 2010.
47 [http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-test-](http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-test-assertions-cd02.pdf)
48 [assertions-cd02.pdf](http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-test-assertions-cd02.pdf)

49 **1.3 Non-normative References**

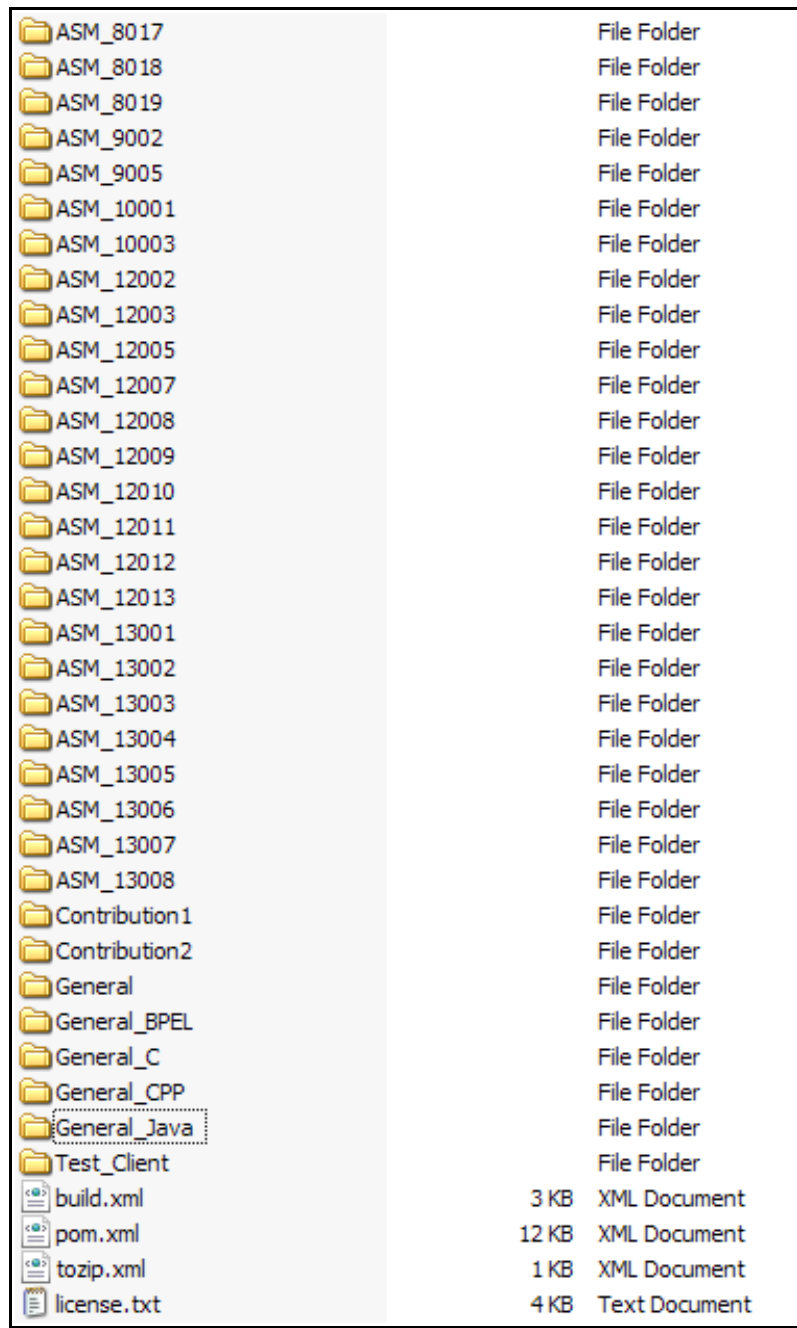
50 None
51

52 2 SCA Assembly Test Suite Organization

53 The SCA Assembly Test Suite is fully described in the TestCases for the SCA Assembly Model
54 specification [SCA-TestCases]. This section describes the aspects of the test suite organization which
55 are relevant when producing a new version of the test suite for a new implementation type.

56 Figure 1 displays part of the main TestCases directory of the SCA Assembly Test Suite.

57



ASM_8017	File Folder
ASM_8018	File Folder
ASM_8019	File Folder
ASM_9002	File Folder
ASM_9005	File Folder
ASM_10001	File Folder
ASM_10003	File Folder
ASM_12002	File Folder
ASM_12003	File Folder
ASM_12005	File Folder
ASM_12007	File Folder
ASM_12008	File Folder
ASM_12009	File Folder
ASM_12010	File Folder
ASM_12011	File Folder
ASM_12012	File Folder
ASM_12013	File Folder
ASM_13001	File Folder
ASM_13002	File Folder
ASM_13003	File Folder
ASM_13004	File Folder
ASM_13005	File Folder
ASM_13006	File Folder
ASM_13007	File Folder
ASM_13008	File Folder
Contribution1	File Folder
Contribution2	File Folder
General	File Folder
General_BPEL	File Folder
General_C	File Folder
General_CPP	File Folder
General_Java	File Folder
Test_Client	File Folder
build.xml	3 KB XML Document
pom.xml	12 KB XML Document
tozip.xml	1 KB XML Document
license.txt	4 KB Text Document

Figure 1: Part of the TestCases Directory of the SCA Assembly Test Suite

59 The contents of the directory are mainly a set of SCA contributions contained in subdirectories such as
60 ASM_8017, ASM_13008 and General. The test client runner application is contained in the Test_Client
61 subdirectory.

62 The General contribution contains a series of artifacts such as composites and WSDL files which are
63 used by a range of testcases.

64 Directories which have names like ASM_8017 contain an SCA contribution with artifacts which are
65 specific to the testcase with the same name as the directory. Typically, testcase-specific contributions
66 are provided for testcases that have artifacts that contain static errors - such contributions are not
67 necessary for testcases which use artifacts that contain no errors.

68 The directories with names like General_BPEL and General_Java are the ones of interest when adapting
69 the test suite to a new implementation type. These directories represent contributions which contain
70 implementation-type dependent artifacts. Their names are stylized so that they have a common stem
71 ("General") followed by an underscore ("_") and then a name which characterizes the implementation
72 type. So "BPEL" is used for the WS-BPEL implementation type, "Java" is used for the Java POJO
73 implementation type and so on.

74 In general, when adapting the test suite to a new implementation type, it is necessary to produce a new
75 contribution similar to General_BPEL and General_Java, containing all the appropriate artifacts for the
76 new implementation type. The contribution needs to have a name which reflects the new implementation
77 type, such as "General_Foo" - the exact name is not important, but it should be unique. The name suffix
78 (e.g. "Foo") is used as part of the configuration of the Test Client.

79 The adapted test suite MUST contain a directory which contains an SCA contribution, termed the
80 "implementation type contribution", holding the implementation-type dependent artifacts, [TST10001]

81 The implementation type contribution must have a name that starts with the stem "General_" followed by
82 a suffix which is unique to the implementation type. [TST10002]

83 2.1 Implementation Type Dependent Artifacts

84 To produce the contributions containing implementation type dependent artifacts, it is necessary to
85 produce:

- 86 • A specific set of composite files which contain components that use the implementation type
- 87 • An associated set of artifacts required to satisfy the set of composite files. This set of artifacts
88 will always contain a set of implementation artifacts in the form used by the implementation type,
89 each supplying a specific componentType. The set of artifacts can also contain other artifacts if
90 required by the implementation type, for example interface definition files.
- 91 • An sca-contribution.xml file which is used to declare SCA exports from this contribution, which
92 are then available for imports to satisfy references to the artifacts made from other contributions

93 It is useful to use the General_Java contribution as the canonical example of what is required for any
94 new contribution that supports a new implementation type.

95 First, the set of implementation-type dependent SCA composite files, which are held in the
96 \src\main\resources directory are shown in Listing 1:

```
97 TestClient_0002.composite  
98 TestClient_0003.composite  
99 TestClient_0004.composite  
100 TestComposite1.composite  
101 TestComposite4.composite  
102 TestComposite5.composite
```

```
103 TestComposite6.composite
104 TestComposite7.composite
105 TestComposite8.composite
106 TestComposite9.composite
107 TestComposite52.composite
108 TestComposite53.composite
109 TestComposite54.composite
110 TestComposite55.composite
111 TestComposite60.composite
112 TestComposite61.composite
113 TestComposite65.composite
114 TestComposite66.composite
115 TestComposite70.composite
116 TestComposite71.composite
117 TestComposite73.composite
```

118 *Listing 1: List of Implementation-Type Dependent Composite files*

119 For Java POJOs, the implementation artifacts consist of:

- 120 • Implementations, which are simple Java classes
- 121 • Java interfaces, which are Java versions of the set of service interfaces used by the SCA
- 122 Assembly Test Suite
- 123 • Exceptions, which are present as a set of Java classes

124 The Java POJO implementations are shown in Listing 2:

```
125 ASM_0002_Client.java
126 ASM_0003_Client.java
127 Service1Callback5Impl.java
128 Service1Callback9Impl.java
129 Service1Impl.java
130 Service1Impl2.java
131 Service1Impl3.java
132 Service1Impl4.java
133 Service1Impl5.java
134 Service1Impl6.java
135 Service1Impl7.java
136 Service1Impl8.java
137 Service1Impl9.java
138 Service1SupersetImpl.java
139 Service2Impl.java
140 Service4Impl.java
141 Service5Impl.java
142 Service5Impl2.java
143 Service9Impl.java
```

144 *Listing 2: List of Implementations*

145 The Java interfaces used in conjunction with the Java POJO implementations are shown in Listing 3:

```
146 Service1.java
147 Service1_Intent.java
148 Service1Superset.java
149 Service2.java
```

```
150 Service4.java
151 Service5.java
152 Service5Callback.java
153 Service8Callback.java
154 Service9.java
155 Service9Callback.java
156 TestInvocation.java
```

158 *Listing 3: List of Implementation-Type Dependent Interface files*

159 The Java exceptions used in conjunction with the Java POJO implementations are shown in Listing 4:

```
160 TestException.java
161 InvalidSCAConfigurationException
```

162 *Listing 4: List of Java Exception Classes*

163 **2.1.1 Implementation-Dependent Composite Files**

164 It is necessary to produce a version of each of the implementation-type dependent composite files shown
165 in Listing 1. Most of these composite files are basically wrappers for a single implementation artifact, and
166 their role is to provide an implementation with a consistent componentType that can be used within
167 higher level composites that are implementation type independent. Typically, the implementation used
168 within each of these composites has the same componentType as the composite itself.

169 As an example of what needs to be done, examine the Java POJO version of TestComposite4, shown in
170 Listing 5:

```
171 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
172           targetNamespace=
173             "http://docs.oasis-open.org/ns/opencsa/scatests/200903"
174           name="TestComposite4">
175
176   <service name="Service1" promote="Composite4Component1/Service1">
177     <interface.java interface="org.oasisopen.sca.test.Service1"/>
178   </service>
179
180   <property name="serviceName" type="string"/>
181
182   <component name="Composite4Component1">
183     <implementation.java class="org.oasisopen.sca.test.Service1Impl2"/>
184     <service name="Service1">
185       <interface.java interface="org.oasisopen.sca.test.Service1"/>
186     </service>
187     <property name="serviceName" source="$serviceName"/>
188     <reference name="reference1"/>
189   </component>
190
191   <reference name="Reference1" promote="Composite4Component1/reference1"
192             multiplicity="1..1">
193     <interface.java interface="org.oasisopen.sca.test.Service1"/>
194   </reference>
195
196 </composite>
```

197 *Listing 5: TestComposite4.composite from General_Java*

198 This corresponds to a componentType with:

- 199 • 1 service with the name "Service1" with the interface Service1

- 200 • 1 reference with the name "Reference1" with the interface "Service1" and multiplicity "1..1"
- 201 • 1 property with the name "serviceName" with the type "xsd:string"

202 A rendering of the componentType of TestComposite4 is shown in Listing 6:

```

203 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
204     targetNamespace=
205     "http://docs.oasis-open.org/ns/opencsa/scatests/200903"
206     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
207     name="ServiceImpl2">
208   <service name="Service1">
209     <interface.java interface="org.oasisopen.sca.test.Service1"/>
210   </service>
211
212   <property name="serviceName" type="xsd:string"/>
213
214   <reference name="reference1">
215     <interface.java interface="org.oasisopen.sca.test.Service1"/>
216   </reference>
217
218 </componentType>

```

219 *Listing 6: ComponentType of TestComposite4.composite*

220 Note that here, the interface type used is interface.java - this can be mapped to interface.wsdl, where the
 221 equivalent interface declaration is shown in Listing 7:

```

222 <interface.wsdl
223   interface="http://test.sca.oasisopen.org/#wsdl.porttype(Service1)"/>

```

224 *Listing 7: Service1 Interface declaration in terms of interface.wsdl*

225 A version of TestComposite4 for the C++ implementation type is shown in Listing 8 with the
 226 corresponding componentType in Listing 9.

```

227 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
228     targetNamespace=
229     "http://docs.oasis-open.org/ns/opencsa/scatests/200903"
230     name="TestComposite4">
231
232   <service name="Service1" promote="Composite4Component1/Service1">
233     <interface.cpp header="test\Service1.h" remotable="true"/>
234   </service>
235
236   <property name="serviceName" type="string"/>
237
238   <component name="Composite4Component1">
239     <implementation.cpp library="general" path="bin\"
240       class="serviceImpl2"/>
241     <service name="Service1">
242       <interface.cpp header="test\Service1.h" remotable="true"/>
243     </service>
244     <property name="serviceName" source="$serviceName"/>
245     <reference name="reference1"/>
246   </component>
247
248   <reference name="Reference1" promote="Composite4Component1/reference1"
249     multiplicity="1..1">
250     <interface.cpp header="test\Service1.h" remotable="true"/>
251   </reference>
252
253 </composite>

```

254 *Listing 8: TestComposite4.composite from General_CPP*

```

255 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
256     targetNamespace=
257     "http://docs.oasis-open.org/ns/opencsa/scatests/200903"

```

```

258         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
259         name="Service1Impl2">
260     <service name="Service1">
261         <interface.cpp header="test\Service1.h" remotable="true" />
262     </service>
263
264     <property name="serviceName" type="xsd:string"/>
265
266     <reference name="reference1">
267         <interface.cpp header="test\Service1.h" remotable="true" />
268     </reference>
269
270 </componentType>

```

271 *Listing 9: Service1Impl2.componentType from General_CPP*

272 The C++ version in Listing 8 illustrates what is necessary to produce a version of TestComposite4 for a
273 new implementation type. Listing 8 has a <implementation.cpp/> element instead of the
274 <implementation.java/> element in Listing 5. To create the corresponding TestComposite4 for
275 "implementation.foo", the <implementation.java/> element shown in Listing 5 has to be replaced by an
276 appropriate <implementation.foo/> element, which needs to reference an implementation artifact that
277 provides a componentType as shown in Listing 6, either implicitly or explicitly as appropriate for the
278 implementation type.

279 If the new implementation type requires the use of an interface type other than WSDL, e.g.
280 "interface.foo", then replace the interface elements in the implementation type-dependent composite files
281 with interface.foo. For example, notice how the <interface.java/> elements in Listing 5 and Listing 6 are
282 replaced with <interface.cpp/> elements in Listing 8 and 9. Implementation type-dependent composites
283 can also use <interface.wsdl/> elements to declare the interfaces of services and references. Note that
284 all the WSDL files declaring the various interfaces can be found in the "General" contribution.

285 This procedure would create a new version of the TestComposite4.composite file as shown in Listing 10:

```

286 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
287     targetNamespace=
288     "http://docs.oasis-open.org/ns/opencsa/scatests/200903"
289     name="TestComposite4">
290
291     <service name="Service1" promote="Composite4Component1/Service1">
292         <interface.foo.../>
293     </service>
294
295     <property name="serviceName" type="string"/>
296
297     <component name="Composite4Component1">
298         <implementation.foo.../>
299         <service name="Service1">
300             <interface.foo.../>
301         </service>
302         <property name="serviceName" source="$serviceName"/>
303         <reference name="reference1"/>
304     </component>
305
306     <reference name="Reference1" promote="Composite4Component1/reference1"
307         multiplicity="1..1">
308         <interface.foo.../>
309     </reference>
310
311 </composite>

```

312 *Listing 10: TestComposite4.composite for implementation type "foo"*

313 The "implementation type contribution" MUST contain a set of composite files as defined in section
314 "Composite Files in the Implementation Type Contribution", each file having the composite name and
315 having the componentType defined there, and where the components in those composite files use
316 implementations of the type supported by the adapted test suite. [TST10003]

317 The composites contained in the "implementation type contribution" MAY use an implementation-type
318 specific form of interface declaration rather than using the interface.wSDL form of interface declaration. [T
319 ST10004]

320 2.1.2 Implementations

321 It is necessary to produce an implementation type specific version of each of the implementation
322 artifacts contained in Listing 2. How this is done depends on the implementation type itself. These
323 implementation artifacts are referenced by the implementation-type dependent composites discussed in
324 the section "Implementation-Dependent Composite Files".

325 Each implementation artifact needs to have a componentType as required by the composite(s) that use
326 it.

327 2.1.3 Interfaces

328 If an implementation type requires the use of a new interface definition type, the adapted test suite
329 MUST contain an interface definition type version of each of the interface artifacts contained in Listing 3.
330 [TST10005] Each interface artifact in the adapted test suite MUST map to equivalent WSDL file of the
331 same name contained in the General contribution. [TST10006]

332 2.1.4 sca-contribution.xml file

333 The General contribution for a new implementation type needs to have an sca-contribution file containing
334 appropriate imports and exports for resolving implementation type specific artifacts.

335 The sca-contribution.xml file from the General_Java contribution is shown in Listing 11:

```
336 <!-- sca-contribution for General_Java contribution -->  
337 <contribution xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912">  
338  
339   <export namespace="http://docs.oasis-open.org/ns/opencsa/scatests/200903"/>  
340   <!-- Composites namespace 1 -->  
341   <export namespace=  
342     "http://docs.oasis-open.org/ns/opencsa/scatests/2009032"/>  
343   <!-- Composites namespace 2 -->  
344   <export.java package="org.oasisopen.sca.test"/>  
345   <!-- Java package -->  
346  
347 </contribution>
```

348 *Listing 11: sca-contribution.xml file from General_Java contribution*

349 It is necessary to export the two composites namespaces shown in Listing 11 – elements defined in
350 these namespaces are used by the composites in the contribution and are therefore necessary if
351 composite references used by the testcases are to resolve correctly to the composites in the contribution.

352 The final export shown in Listing 11 is specific to the Java POJO implementation type - it enables the re-
353 use of Java implementations and Java interface artifacts by other contributions in the test suite. Whether
354 this is relevant to a new implementation type depends first on whether that implementation type has an
355 implementation type specific artifact sharing mechanism. It also depends on whether it makes sense to
356 share artifacts in the implementation type General contribution with other language-specific
357 contributions.

358 2.2 The Test Client

359 The SCA Assembly Test Suite contains a Test Client that is used to start and run the testcases. The test
360 client is responsible for:

- 361 • Starting the SCA Runtime that is under test
- 362 • Loading the set of contributions relevant to the specific testcase into the SCA Runtime
- 363 • Invoking the test
- 364 • Stopping/unloading the contributions related to the testcase

365 The SCA Assembly Test Suite supplies a Test Client that is written using Java - it depends only on the
366 capabilities of the publicly available JavaSE runtime. Facilities are provided in the test client for a piece
367 of SCA Runtime-dependent code to be provided by the SCA Runtime provider, which has the task of
368 starting the SCA Runtime and of passing to it the contributions and artifacts used for testcases. This
369 permits the test client to be adapted to any particular SCA Runtime.

370 The test client invokes the testcase using Web services. The testcases are all built to expose a Web
371 service endpoint that can be used to invoke the test and which feeds back the results of executing the
372 test. This separates the test client from the SCA Runtime and it means that the SCA Runtime can in
373 principle be implemented using any technology, independent of the test client. Thus it is possible to use
374 the test client supplied with the SCA Assembly Test Suite for a wide variety of SCA runtimes.

375 The test client contains two sets of configuration information:

- 376 • General configuration of the test client itself that applies to all testcases
- 377 • Testcase-specific configuration that describes the important aspects of each testcase

378 2.2.1 Configuring the Test Client

379 The Java test client has some general configuration that can be used to influence the way that the test
380 client operates. This general configuration is contained in the properties file "oasis-sca-tests.properties".
381 An example of this properties file is shown in Listing 12:

```
382 #  
383 # Copyright (C) OASIS (R) 2005,2009. All Rights Reserved.  
384 # OASIS trademark, IPR and other policies apply.  
385 #  
386  
387 # OASIS SCA Assembly test properties  
388 #  
389 # 1) The implementation type to use for Assembly test suite  
390 # Examples: "Java" "BPEL" "CPP" "C"  
391 org.oasis.sca.tests.assembly.lang=Java  
392  
393 # 2) The class to use as the Runtime Bridge for the SCA runtime under test  
394 org.oasis.sca.tests.assembly.runtime_bridge=client.TuscanyRuntimeBridge  
395  
396 # 3) The location of the contributions for the test suite  
397 # %1 represents the placement of the name of each contribution into the  
398 # location URI  
399 # Following uses ZIP files for the contributions  
400 org.oasis.sca.tests.assembly.contribution.location=file:/C:/OASIS_TESTS/SCA-  
401 Assembly/TestCases/%1/target/%1.zip
```

402 *Listing 12: Example of the oasis-sca-tests.properties file*

403 The oasis-sca-tests.properties file contains 3 pieces of configuration:

- 404 1. The name of the implementation type to use (e.g. "Java")
- 405 2. The name of the runtime bridge to use (i.e. configures the client to use a particular SCA
- 406 Runtime)
- 407 3. The location to use for the contributions at runtime, supplied as a URL with a substitution string
- 408 for the short name of the contribution:
- 409 file:/C:/OASIS_TESTS/SCA-Assembly/TestCases/%1/target/%1.zip
- 410 where %1 is substituted at runtime with the name of the contribution
- 411 This provides flexibility in the location of the contributions.

412 The runtime bridge is a piece of code that connects the test client to the SCA Runtime that is used to run

413 the testcases. It is described in detail in the TestCases documentation for SCA Assembly [SCA-

414 TestCases]. The configuration needs to contain the fully qualified name of the class for the runtime

415 bridge. So, for a new implementation type, it is necessary to have a runtime bridge for the SCA Runtime

416 that provides that implementation type and for the name of the runtime bridge to be entered into the

417 properties file.

418 To use a different runtime bridge, the Java classes for the bridge are added to a (sub)directory of the test

419 client code and the name of the main class for the bridge is put into the properties file.

420 The name of the implementation type, such as "Java" needs to match the name used for the suffix of the

421 implementation-type dependent contributions described in the section "[Implementation Type Dependent](#)

422 [Artifacts](#)". This name is used to influence the detailed configuration for each testcase, described in the

423 next section.

424 2.2.2 Configuration of Each TestCase

425 For the supplied test client, the configuration of each testcase is contained in the Test_Client

426 subdirectory of the test suite, specifically in the ASM_nnnn_TestCase.java files contained in the

427 src/main/javaclient subdirectory. Listing 10 shows the contents of a typical client file,

428 ASM_5004_TestCase.java:

```
429     protected TestConfiguration getTestConfiguration() {
430         TestConfiguration config = new TestConfiguration();
431         config.testName         = this.getClass().getSimpleName().substring(0, 8);
432         config.input            = "request";
433         config.output[0]       = "exception";
434         config.composite        = "Test_" + config.testName + ".composite";
435         config.testServiceName = "TestClient";
436
437         config.contributionNames =
438             new String[] { "ASM_5004", "General", "General" + _Lang };
439         config.serviceInterface = TestInvocation.class;
440         return config;
441     }
```

442 *Listing 10: Contents of Test Client configuration for ASM_5004_TestCase*

443 The significant aspects of the configuration are:

- 444 1. The test name = "ASM_5004"
- 445 2. Input string = "request"
- 446 3. Expected output string = "exception"
- 447 Note - the string can be some string like "ASM_5002 request service1 operation1 invoked" if the
- 448 testcase is expected to succeed (i.e., it is a positive test), but the string "exception" is used
- 449 where it is expected that the test will fail (i.e., it is a negative test) and the SCA Runtime is
- 450 expected to raise an exception.

- 451 4. Composite to run = "Test_ASM_5004.composite"
452 Every testcase involves running one specific composite, which is present in one of the
453 contributions - this name is supplied in the configuration.
- 454 5. ContributionNames = "ASM_5004", "General", "General" + _Lang
455 This is a list or array of the names of the Contributions to use for this testcase. The test client
456 MUST load all the contributions declared in the testcase configuration into the SCA Runtime. [TS
457 T10007] Note that the final one in this list is a contribution which depends on the name of the
458 implementation type that is under test - "Lang" gets replaced with the name of the
459 implementation type, to result in a contribution name like "General_Java" or "General_BPEL",
460 based on the implementation type name contained in the general configuration of the test client.
- 461 6. ServiceInterface = "TestInvocation"
462 In principle the test client allows for different Web service interfaces to be used to communicate
463 from the test client to the SCA application. This piece of the configuration names the interface to
464 use. However, in practice, for the SCA Assembly Test Suite, only a single interface is used for all
465 of the testcases - "TestInvocation".

466 2.2.3 Creating an Alternative Test Client

467 If it is impossible or difficult to use the Java test client supplied with the test suite, it is possible to create
468 a new test client written using some alternative technology. The main requirements are that:

- 469 • The test client SHOULD execute the testcases through Web services. [TST10008] , The Java
470 test client is a good example of this approach. This is to ensure separation of the client from the
471 SCA Runtime that is being tested.
- 472 • The test client MUST have a means of starting the SCA Runtime, a means of passing the test
473 artifacts to the SCA Runtime (contributions and test composite) and a means of starting the SCA
474 application (based on the test composite) [TST10009]
- 475 • The test client MUST have a mechanism for holding the configuration of each testcase defined in
476 the test suite and of using that configuration when called on to execute a particular testcase. [TS
477 T10010]
- 478 • The test client MUST be able to compare the expected output of the test with the actual output,
479 including those testcases that are expected to fail with an exception. The test client MUST
480 report the success or failure of the testcase dependant on whether the expected output matches
481 the actual output. [TST10011]
482 (Note that in the case of testcases which are expected to raise an exception, SCA makes no
483 statement concerning what exception is raised - only that some exception is raised)

484

485

486

487

488

3 Conformance

489 For an adapted version of the SCA Assembly Test Suite that claims to conform to the requirements of
490 this specification MUST meet the following conditions

491 1. The adapted version of the test suite MUST comply with all the mandatory statements listed in in
492 the table Mandatory Items in the appendix "Conformance Items"

493

494 A. Required Artifacts

495

496 A.1. Composite Files in the Implementation Type 497 Contribution

498 The General_xxx implementation type contribution contains composite file with the following names:

- 499 • TestClient_0002
- 500 • TestClient_0003
- 501 • TestClient_0004
- 502 • TestComposite1
- 503 • TestComposite4
- 504 • TestComposite5
- 505 • TestComposite6
- 506 • TestComposite7
- 507 • TestComposite8
- 508 • TestComposite9
- 509 • TestComposite52
- 510 • TestComposite53
- 511 • TestComposite54
- 512 • TestComposite55
- 513 • TestComposite60
- 514 • TestComposite61
- 515 • TestComposite65
- 516 • TestComposite66
- 517 • TestComposite70
- 518 • TestComposite71
- 519 • TestComposite73

520 The componentType of the composites is as shown in the following listings:

521

```
522 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
523             xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
524   <service name="TestInvocation">  
525     <interface.wSDL  
526       interface="http://test.sca.oasisopen.org/  
527               #wsdl.porttype(TestInvocation)"/>
```

```

528     </service>
529
530     <property name="testName" type="xsd:string"/>
531
532     <reference name="reference1" multiplicity="1..1">
533         <interface.wSDL
534             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
535     </reference>
536
537 </componentType>

```

538 *Listing A1: ComponentType of TestClient_0002 composite*

539

```

540 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
541               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
542     <service name="TestInvocation">
543         <interface.wSDL
544             interface="http://test.sca.oasisopen.org/
545                     #wsdl.porttype (TestInvocation)"/>
546     </service>
547
548     <property name="testName" type="xsd:string"/>
549
550     <reference name="reference1" multiplicity="1..n">
551         <interface.wSDL
552             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
553     </reference>
554
555 </componentType>

```

556 *Listing A2: ComponentType of TestClient_0003 composite*

557

```

558 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
559               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
560     <service name="TestInvocation">
561         <interface.wSDL
562             interface="http://test.sca.oasisopen.org/
563                     #wsdl.porttype (TestInvocation)"/>
564     </service>
565
566     <property name="testName" type="xsd:string"/>
567
568     <reference name="reference1" multiplicity="0..1">
569         <interface.wSDL
570             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
571     </reference>
572
573 </componentType>

```

574 *Listing A3: ComponentType of TestClient_0004 composite*

575

```

576 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
577               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
578     <service name="Service1">
579         <interface.wSDL
580             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
581     </service>
582
583     <property name="serviceName" type="xsd:string"/>
584
585 </componentType>

```

586 *Listing A4: ComponentType of TestComposite1 composite*

587

```
588 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
589     xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
590   <service name="Service1">  
591     <interface.wSDL  
592       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>  
593   </service>  
594  
595   <property name="serviceName" type="xsd:string"/>  
596  
597   <reference name="Reference1" multiplicity="1..1">  
598     <interface.wSDL  
599       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>  
600   </reference>  
601 </componentType>  
602
```

603 *Listing A5: ComponentType of TestComposite4 composite*

604

```
605 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
606     xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
607   <service name="Service1">  
608     <interface.wSDL  
609       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service2) "/>  
610   </service>  
611  
612   <property name="serviceName" type="xsd:string"/>  
613 </componentType>  
614
```

615 *Listing A6: ComponentType of TestComposite5 composite*

616

```
617 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
618     xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
619   <service name="Service1">  
620     <interface.wSDL  
621       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>  
622   </service>  
623  
624   <property name="serviceName" type="xsd:string"/>  
625  
626   <reference name="Reference1" multiplicity="1..n">  
627     <interface.wSDL  
628       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>  
629   </reference>  
630 </componentType>  
631
```

632 *Listing A7: ComponentType of TestComposite6 composite*

633

```
634 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"  
635     xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
636   <service name="Service1">  
637     <interface.wSDL  
638       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>  
639   </service>  
640  
641   <property name="serviceName" type="xsd:string"/>  
642  
643   <reference name="Reference1" multiplicity="0..n">  
644     <interface.wSDL  
645       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>
```

```
646     </reference>
647
648 </componentType>
```

649 *Listing A8: ComponentType of TestComposite7 composite*

```
650
651 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
652               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
653   <service name="Service1">
654     <interface.wSDL
655       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
656   </service>
657
658   <property name="serviceName" type="xsd:string"/>
659
660   <reference name="Reference1" multiplicity="0..1">
661     <interface.wSDL
662       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
663   </reference>
664 </componentType>
```

666 *Listing A9: ComponentType of TestComposite8 composite*

```
667
668 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
669               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
670   <service name="Service1Superset">
671     <interface.wSDL
672       interface="http://test.sca.oasisopen.org/
673               #wsdl.porttype (Service1Superset)"/>
674   </service>
675
676   <property name="serviceName" type="xsd:string"/>
677 </componentType>
```

679 *Listing A10: ComponentType of TestComposite9 composite*

```
680
681 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
682               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
683   <service name="Service1">
684     <interface.wSDL
685       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service4)"/>
686   </service>
687
688   <property name="serviceName" type="xsd:string"/>
689 </componentType>
```

691 *Listing A11 ComponentType of TestComposite52 composite*

```
692
693 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
694               xmlns:xsd="http://www.w3.org/2001/XMLSchema">
695   <service name="Service1">
696     <interface.wSDL
697       interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
698   </service>
699
700   <property name="serviceName" type="xsd:string"/>
701
702   <reference name="Reference1" multiplicity="1..1">
703     <interface.wSDL
```

```

704         interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service4) "/>
705     </reference>
706
707 </componentType>

```

708 *Listing A12: ComponentType of TestComposite53 composite*

```

709
710 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
711     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
712     <service name="Service1">
713         <interface.wsdl
714             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>
715     </service>
716
717     <property name="serviceName" type="xsd:string"/>
718
719     <reference name="Referencel" multiplicity="1..1">
720         <interface.wsdl
721             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service5) "
722             callbackInterface="http://test.sca.oasisopen.org/
723                 #wsdl.porttype (Service5Callback) "/>
724     </reference>
725
726 </componentType>

```

727 *Listing A13: ComponentType of TestComposite54 composite*

```

728
729 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
730     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
731     <service name="Service1">
732         <interface.wsdl
733             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service5) "
734             callbackInterface="http://test.sca.oasisopen.org/
735                 #wsdl.porttype (Service5Callback) "/>
736     </service>
737
738     <property name="serviceName" type="xsd:string"/>
739
740 </componentType>

```

741 *Listing A14 ComponentType of TestComposite55 composite*

```

742
743 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
744     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
745     <service name="Service1">
746         <interface.wsdl
747             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>
748     </service>
749
750     <property name="serviceName" type="xsd:string"/>
751
752     <reference name="Referencel" multiplicity="1..1">
753         <interface.wsdl
754             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service9) "/>
755     </reference>
756
757 </componentType>

```

758 *Listing A15: ComponentType of TestComposite60 composite*

```

759
760 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
761     xmlns:xsd="http://www.w3.org/2001/XMLSchema">

```

```

762     <service name="Service1">
763         <interface.wSDL
764             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service9) "/>
765     </service>
766
767     <property name="serviceName" type="xsd:string"/>
768
769 </componentType>

```

770 *Listing A16 ComponentType of TestComposite61 composite*

771

```

772 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
773     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
774     <service name="Service1">
775         <interface.wSDL
776             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service5) "
777             callbackInterface="http://test.sca.oasisopen.org/
778                 #wsdl.porttype (Service5Callback) "/>
779     </service>
780
781     <property name="serviceName" type="xsd:string"/>
782
783 </componentType>

```

784 *Listing A17 ComponentType of TestComposite65 composite*

785

```

786 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
787     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
788     <service name="Service1">
789         <interface.wSDL
790             interface="http://test.sca.oasisopen.org/
791                 #wsdl.porttype (Service1_Intent) "/>
792     </service>
793
794     <property name="serviceName" type="xsd:string"/>
795
796 </componentType>

```

797 *Listing A18 ComponentType of TestComposite66 composite*

798

```

799 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
800     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
801     <service name="Service1">
802         <interface.wSDL
803             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1) "/>
804     </service>
805
806     <property name="serviceName" type="xsd:string"/>
807
808     <reference name="Reference1" multiplicity="1..1">
809         <interface.wSDL
810             interface="http://test.sca.oasisopen.org/
811                 #wsdl.porttype (Service1Superset) "/>
812     </reference>
813
814 </componentType>

```

815 *Listing A19: ComponentType of TestComposite70 composite*

816

```

817 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
818     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
819     <service name="Service1">

```

```
820     <interface.wSDL
821         interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
822     </service>
823
824     <property name="serviceName" type="xsd:string"/>
825     <property name="serviceData1" type="xsd:string"/>
826     <property name="serviceData2" type="xsd:string"/>
827
828 </componentType>
```

829 *Listing A20 ComponentType of TestComposite71 composite*

830

```
831 <componentType xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
832     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
833     <service name="Service1">
834         <interface.wSDL
835             interface="http://test.sca.oasisopen.org/#wsdl.porttype (Service1)"/>
836     </service>
837
838     <property name="serviceName" type="xsd:string"/>
839     <property name="serviceData1" type="xsd:float"/>
840
841 </componentType>
```

842 *Listing A21 ComponentType of TestComposite73 composite*

843

844

B. Conformance Items

845

This section contains a list of conformance items for the SCA Assembly Test Suite Adaptation specification.

846

847

B.1. Mandatory Items

Conformance ID	Description
[TST10001]	The adapted test suite MUST contain a directory which contains an SCA contribution, termed the "implementation type contribution", holding the implementation-type dependent artifacts,
[TST10002]	The implementation type contribution must have a name that starts with the stem "General_" followed by a suffix which is unique to the implementation type.
[TST10003]	The "implementation type contribution" MUST contain a set of composite files as defined in section "Composite Files in the Implementation Type Contribution", each file having the composite name and having the componentType defined there, and where the components in those composite files use implementations of the type supported by the adapted test suite.
[TST10004]	The composites contained in the "implementation type contribution" MAY use an implementation-type specific form of interface declaration rather than using the interface.wsdl form of interface declaration.
[TST10005]	If an implementation type requires the use of a new interface definition type, the adapted test suite MUST contain an interface definition type version of each of the interface artifacts contained in Listing 3.
[TST10006]	Each interface artifact in the adapted test suite MUST map to equivalent WSDL file of the same name contained in the General contribution.
[TST10007]	The test client MUST load all the contributions declared in the testcase configuration into the SCA Runtime.
[TST10008]	The test client SHOULD execute the testcases through Web services
[TST10009]	The test client MUST have a means of starting the SCA Runtime, a means of passing the test artifacts to the SCA Runtime (contributions and test composite) and a means of starting the SCA application (based on the test composite)
[TST10010]	The test client MUST have a mechanism for holding the configuration of each testcase defined in the test suite and of using that configuration when called on to execute a particular testcase.
[TST10011]	The test client MUST be able to compare the expected output of the test with the actual output, including those testcases that are expected to fail with an exception. The test client MUST report the success or failure of the testcase dependant on whether the expected output matches the actual output.

848

C. Acknowledgments

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

851 **Participants:**

- 852 • Mike Edwards, IBM
- 853 • Dave Booz, IBM
- 854 • Bryan Aupperle, IBM
- 855 • Jeff Estefan, Jet Propulsion Laboratory

856

D. Revision History

857

Revision	Date	Editor	Changes Made
1	04/19/10	Mike Edwards	Initial version created
2	04/22/10	Mike Edwards	Updated and extended based on feedback
3	05/12/10	Jeff Estefan	Clean-up and formatting consistency
4	05/13/10	Mike Edwards	Initial set of normative statements Appendix for normative statements added Appendix for General _xxx contents added
5	06/15/10	Mike Edwards	Added line numbering
cd01	06/20/10	Mike Edwards	All changes accepted.

858