TestCases for the SCA-J Common Annotations and APIs Specification Version 1.1 ~~J Common Annotations and APIs Version 1.1 Specification~~

Committee Specification Draft 02 / Public Review Draft 02~~Draft 01~~

24 October 2011

Specification URIs

This version:

http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-csprd02.pdf (Authoritative)
http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-csprd02.html
http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-csprd02.odt

Previous version:

http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-cd01.pdf (Authoritative)

~~19 July 2010~~

~~**Specification URIs:**~~

~~**This Version:**~~

http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-cd01.html

http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-cd01.odt

Latest version:

http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases.pdf (Authoritative)
~~http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-cd01.pdf(Authoritative)~~

~~**Previous Version:**~~

~~N/A~~

~~**Latest Version:**~~

http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases.html
http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases.odt
~~http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases.pdf (Authoritative)~~

Technical Committee:
OASIS Service Component Architecture / J (SCA-J) TC
~~http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=sca-j~~

Chairs~~(s)~~:

David Booz (booz@us.ibm.com), IBM~~, IBM~~

~~Mark Combellack Avaya Inc~~

Anish Karmarkar (Anish.Karmarkar@oracle.com), Oracle

Editors:

Mike Edwards (mike_edwards@uk.ibm.com), IBM

David Booz (booz@us.ibm.com), IBM

Additional artifacts:

This prose specification is one component of a Work Product which also includes:

- Test suite artifacts ZIP archive: http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-csprd02/testcases.zip

Related work:

This specification is related to:

- *Service Component Architecture SCA-J Common Annotations and APIs Specification Version 1.1*. Latest version. http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec.html

Declared XML namespaces:

~~**Editor(s):**~~

~~Mike Edwards, IBM~~

~~David Booz, IBM~~

~~**Related Work:**~~

~~This document is related to:~~

- ~~Service Component Architecture Java Common Annotations and APIs Specification Version 1.1~~
  ~~http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-1.1-spec-cd04.pdf~~

~~**Declared XML Namespace(s):**~~

http://docs.oasis-open.org/ns/opencsa/scatests/200903

- http://docs.oasis-open.org/ns/opencsa/scatests/2009032
  ~~http://test.sca.oasisopen.org/~~
- http://test.sca.oasisopen.org/

Abstract:

This document defines the TestCases for the SCA Java Common Annotations and APIs specification.~~Assembly specification.~~

~~The TestCases represent a series of tests that an SCA runtime must pass in order to claim conformance to the requirements of the SCA Java Common Annotations and APIsAssembly specification.~~

The TestCases represent a series of tests that an SCA runtime must pass in order to claim conformance to the requirements of the SCA Java Common Annotations and APIs specification.

Status:

This document was last revised or approved by the OASIS Service Component Architecture / J (SCA-J) TC on the above date. The level of approval is also listed above. Check the "Latest version" location noted above for possible later revisions of this document.~~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-j/~~

Technical Committee members should send comments on this Work Product 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-j/.

For information on whether any patents have been disclosed that may be essential to implementing this Work Product, 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-j/ipr.php).

Citation format:

When referencing this Work Product the following citation format should be used:

[SCA-J-CAA-Testcases-V1.1]

TestCases for the SCA-J Common Annotations and APIs Specification Version 1.1. 24 October 2011. OASIS Committee Specification Draft 02 /Public Review Draft 02. http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-testcases-csprd02.html.

~~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-j/ipr.php.~~

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

Notices

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 name "OASIS" is a trademarks "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

This document defines the TestCases for the SCA_J Common Annotations and APIs Version 1.1 S~~As-sembly s~~pecification.

The tests described in this document are derived from the normative statements in SCA_J Common Annotations and APIs Version 1.1 Specification via Test Assertions which are described in Appendix A, "Test Assertions for the SCA Java Common Annotations and APIs Specification Version 1.1"~~related to the Test Assertions described in the SCA Assembly Test Assertions document [CAA-TA]~~.

1.1 TestCase Structure

The SCA J CAA testcases follow a standard structure. They are divided into two main parts:

1. Test Client, which drives the test and checks that the results are as expected
2. Test Application, which forms the bulk of the testcase and which consists of Composites, WSDL files, XSDs and code artifacts such as Java classes, organized into a series of SCA contributions

The basic idea is that the Test Application runs on the SCA runtime that is under test, while the Test Client runs as a standalone application, invoking the Test Application through one or more service interfaces.

Test Client

The test client is designed as a standalone application. The version built here is a Java application which uses the JUnit test framework, although in principle, the client could be built using another implementation technology.

The test client is structured to contain configuration information about the testcase, which consists of:

1. metadata identifying the Test Application in terms of the SCA Contributions that are used and the Composites that ar~~must b~~e deployed and run
2. data indicating which service operation(s) is~~must be~~ invoked with input data and expected output data (including exceptions for expected failure cases)

The Java test client consists of a base runtime class, BaseJAXWSTestCase.java. Each actual testcase is implemented by a small class which extends the base runtime class. The bulk of the code required to run a test is held in the base runtime class. The small testcase class contains the configuration for the specific test, which it provides to the code in the base runtime class through a standard interface.

The Java test client base runtime class is structured so that there is a replaceable class called the RuntimeBridge, which is used to communicate with the SCA runtime under test, for the purposes of deploying and running the test application. Each SCA runtime provider can produce a version of this class. The code within the runtime bridge is likely to be highly proprietary and specific to the SCA runtime for which it is written. Which runtime bridge class is used at runtime is controlled by an environment variable or system variable with the name "OASIS_TESTENV_RUNTIME_BRIDGE_CLASS", which is read by the code in BaseJAXWSTestCase.

The Test Client defaults to using Web services to communicate with the test application. The client is structured to permit Web services to be replaced by some other binding (eg JMS) should the SCA runtime under test not support Web services as a binding technology.

Test Application

Each Test Application consists of one top level SCA Composite file and one or more other SCA Composite files and their associated artifacts (implementations, interface files), plus test client invocation application described above.

A typical test application has a design where the top level composite offers a single service to the client application over a Web services binding. The top level composite contains one component which offers the service that is used by the client application. The top level composite then contains one or more other components which are used by the first component.

All of the components in the top level composite are implemented by composites. These second level composites then contain typically one component, implemented using a specific technology such as Java POJO. In some cases the implementation is~~may be~~ a third level composite.

The application is structured so that alternative technologies can be used. For example, replacing the contents of the second-level or third-level composites allows different Java implementation technologies to be tested – eg POJOs or Spring Application Contexts can be used. Similarly, the binding used to connect from the top level composite to the client application can~~may be used. Similarly, the binding used to connect from the top level composite to the client application may~~ be changed from Web services to JMS if required, simply by changing the binding on the <service/> of the top level composite.

Which implementation language to use for test artifacts is controlled by a system variable or environment variable which is read by the test client application, with the name "OASIS_TESTENV_IMPL_LANG". This variable can have one of the following values:

- "POJO" - for Java implementations
- "Spring" - for Spring implementations

The testcases are designed so that the range of implementation types can be expanded

Test Artifacts Organization

Note that the design of these testcases promotes reuse of artifacts between testcases, so that many testcases share components. For example, components implementing simple invokable services are all implemented using a single parameterized implementation artifact.

All the test artifacts are contained in a number of Contributions, which are simply filesystem directories which are all peers in the filesystem hierarchy. The names of the directories are the names of the Contributions and the names are significant. The names of Contributions containing implementation type specific artifacts (such as Java classes) are also specially structured to allow for replacement of one type of implementation artifact with another.

Broadly, Contribution names are as follows:

- JCA_nnnn - a contribution that is specific for a particular testcase, where "nnnn" is the number of the testcase. Often this is required because a particular testcase involves artifacts that contain errors that are statically checkable - an SCA runtime is permitted to reject such artifacts when they are contributed and deployed and it is important to ensure that contributions containing deliberate errors for one testcase do not interfere with the operation of other testcases.
- JCA_nnnn_POJO - a contribution for a specific testcase where there is a need for language specific artifacts that relate to that testcase alone
- JCA_General - a shared contribution containing implementation type independent artifacts that can be used by many testcases.
- JCA_General_POJO - a shared contribution containing implementation type dependent artifacts for Java POJOs. These artifacts can include both Java classes and also SCA composites that directly use Java classes.

Note that the names of Contributions containing implementation specific artifacts ends with a name that is specific to the implementation type - so "_POJO" is used for Java POJO implementations, "_Spring" is used for Spring implementations (and so on). Note that the name following the underscore matches the name used in the "OASIS_TESTENV_IMPL_LANG" variable used to control execution of the test client. The concept is that where there is an implementation type specific contribution, each implementation type provides its own versions of the same basic artifacts. Typically, this means that each contribution contains~~must provide its own versions of the same basic artifacts. Typically, this means that each contribution must contain~~ the same set of Composites, but that the implementation type dependent artifacts that these composites use will differ from implementation type to implementation type.

Basically, the setting of the variable is used to select the suffix used for implementation type dependent contributions. If the variable is set to "POJO" then the contribution "JCA_General_POJO" is selected, whereas if the variable is set to "Spring", the contribution "JCA_General_Spring" is selected.

1.2 Namespaces and Java Package Names

The SCA Assembly testcase suite makes use of some XML namespaces and Java package names, as follows:

SCA Artifact Namespaces

These apply to artifacts such as Composites

http://docs.oasis-open.org/ns/opencsa/scatests/200903

http://docs.oasis-open.org/ns/opencsa/scatests/2009032

WSDL Namespace

http://test.sca.oasisopen.org/

Java Package name

For Java interface classes and for Java implementation classes

`org.oasisopen.sca.test`

1.3 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]

1.4 Normative References

[RFC 2119] S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. IETF RFC 2119, March 1997. http://www.ietf.org/rfc/rfc2119.txt.

~~[CAA-TA] OASIS Committee Draft 01, "Test Assertions for the SCA_J Common Annotations and APIs Version 1.1 Specification", July 2010. http://docs.oasis-open.org/opencsa/sca-j/sca-j-caa-1.1-test-assertions-cd01.pdf~~

[TA-GUIDE] OASIS Committee Draft 04, *Test Assertion Guidelines*, February 2010. http://docs.oasis-open.org/tag/guidelines/v1.0/cd04/testassertionsguidelines-cd-04.pdf

[JAVACAA] OASIS Committee Specification Draft 06, "Service Component Architecture SCA-J Common Annotations and APIs Specification Version 1.1",

August 2011
http://docs.oasis-open.org/opencsa/sca-j/sca-javacaa-spec-v1.1-csd06.pdf

1.5 Non-normative References

N/A

2 TestCases

2.1 Assembly

JCA_1001_TestCase

Testcase ID	JCA_1001_TestCase
Test Assertion	ASM-TA-8005
Description	Tests that when an <interface/> element of a <component/> <reference/> has an interface that is marked bidirectional, where the forward interface is marked remotable, then the callback interface is also marked remotable
Artifacts	JCA_1001_TestCase.java Test_JCA_1001.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceRemoteLocal.java ServiceRemoteLocalCallback.java service1RemoteLocalCallbackImpl.java serviceRemoreLocalImpl.java
Expected output	Negative test: "exception"

JCA_1002_TestCase

Testcase ID	JCA_1002_TestCase
Test Assertion	ASM-TA-8006
Description	Tests that when an <interface/> element of a <component/> <reference/> has an interface that is marked bidirectional, where the forward interface is marked local, then the callback interface is also marked local
Artifacts	JCA_1002_TestCase.java Test_JCA_1002.composite TestInvocation.wsdl TestClient_0002.composite

	Service1.java ServiceLocalRemote.java ServiceLocalRemoteCallback.java service1LocalRemoteCallbackImpl.java serviceLocalRemoteImpl.java
Expected output	Negative test: "exception"

2.2 Section 2

JCA_2001_TestCase

Testcase ID	JCA_2001_TestCase
Test Assertion	JCA-TA-2001, ASM-TA-8001
Description	Tests that a Java service which is marked remotable does not use method overloading
Artifacts	JCA_2001_TestCase.java Test_JCA_2001.composite TestInvocation.wsdl TestClient_0002.composite Service1Overload.java service1OverloadImpl.java
Expected output	Negative test: “exception”
~~Testcase ID~~	~~JCA_2001_TestCase~~
~~Test Assertion~~	~~JCA-TA-2001~~
~~Description~~	~~Tests that a Java service which is marked remotable does not use method overloading~~
~~Artifacts~~	~~JCA_2001_TestCase.java~~ ~~Test_JCA_2001.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1Overload.java~~ ~~service1OverloadImpl.java~~
~~Expected output~~	~~Negative test:~~ ~~“exception”~~

JCA_2002_TestCase

Testcase ID	JCA_2002_TestCase
Test Assertion	JCA-TA-2002
Description	Tests that a stateless scoped Java implementation instance is not dispatched on more than a single thread at one time
Artifacts	JCA_2002_TestCase.java Test_JCA_2002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ParallelService.java parallelServiceClientImpl.java parallelServiceImpl.java
Expected output	Positive test: "JCA_2002 request service1 parallel service invocation successful"

131

132 JCA_2003_TestCase

133

Testcase ID	JCA_2003_TestCase
Test Assertion	JCA-TA-2003
Description	Tests that an implementation instance annotated with @Scope("STATELESS") is invoked only once through one business method during the lifetime of the implementation instance
Artifacts	JCA_2003_TestCase.java Test_JCA_2003.composite TestInvocation.wsdl TestClient_0002.composite MultipleService.java multipleServiceClientImpl.java multipleServiceImpl.java
Expected output	Positive test: "JCA_2003 request service1 multiple service invocation successful"

134

135 JCA_2004_TestCase

136

Testcase ID	JCA_2004_TestCase
Test Assertion	JCA-TA-2004
Description	Tests that where there is a Domain-level component implementation marked as COMPOSITE scope, that all its clients appear to interact with a single runtime instance of the class.
Artifacts	JCA_2004_TestCase.java Test_JCA_2004.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1CoordinatorImpl.java service1Impl2.java service1CompositeImpl.java
Expected output	Positive test: "JCA_2004 request serviceCoordinator operation1 invoked service1 operation1 invoked serviceComposite operation1 Initial service2 operation1 invoked serviceComposite operation1 1 service3 operation1 invoked serviceComposite operation1 2 service4 operation1 invoked serviceComposite operation1 3"

137

138 ## JCA_2005_TestCase

139

Testcase ID	JCA_2005_TestCase
Test Assertion	JCA-TA-2005
Description	Tests that where a component implementation is marked with COMPOSITE scope and with @EagerInit, that the implementation instance is created and initialized when the component is started.
Artifacts	JCA_2005_TestCase.java Test_JCA_2005.composite TestInvocation.wsdl TestClient_0002.composite Service1.java compositeEagerInitImpl.java service1CompositeImpl.java
Expected output	Positive test: "JCA_2005 request serviceComposite2 operation1 EagerInit called"

140

141 JCA_2006_TestCase

142

Testcase ID	JCA_2006_TestCase
Test Assertion	JCA-TA-2006, JCA-TA-10065
Description	Tests that where a component implementation has a method marked with @Init, that this method is called when the implementation instance is created
Artifacts	JCA_2006_TestCase.java Test_JCA_2006.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1InitCheckerImpl.java service1InitImpl.java
Expected output	Positive test: "JCA_2006 request serviceComposite1 operation1 Init check succeeded"
~~Testcase ID~~	~~JCA_2006_TestCase~~
~~Test Assertion~~	~~JCA-TA-2006~~
~~Description~~	~~Tests that where a component implementation has a method marked with @Init, that this method is called when the implementation instance is created~~
~~Artifacts~~	~~JCA_2006_TestCase.java~~ ~~Test_JCA_2006.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.java~~ ~~service1InitCheckerImpl.java~~ ~~service1InitImpl.java~~
~~Expected output~~	~~Positive test:~~ ~~"JCA_2006 request serviceComposite1 operation1 Init check succeeded"~~

143

144 JCA_2007_TestCase

145

Testcase ID	JCA_2007_TestCase
Test Assertion	JCA-TA-2007
Description	Tests that for a Java implementation with COMPOSITE scope, that multiple invocations of service operations which overlap in time can run within a single instance of the implementation on separate Java threads.
Artifacts	JCA_2007_TestCase.java

	Test_JCA_2007.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ParallelService.java parallelCompositeClientImpl.java parallelCompositeServiceImpl.java
Expected output	Positive test: "JCA_2007 request serviceCompositeClient COMPOSITE service invocation successfully used by multiple threads simultaneously"

146

147 ## JCA_2008_TestCase

148

Testcase ID	JCA_2008_TestCase
Test Assertion	JCA-TA-2008
Description	Tests that for a Java implementation class with COMPOSITE scope offering a service, used as the implementation of a component within a composite which is itself the implementation of a Domain level component, all clients of the component service appear to interact with a single instance of the implementation class
Artifacts	JCA_2008_TestCase.java Test_JCA_2008.composite TestInvocation.wsdl TestClient_0002.composite CompositeScope.composite Service1.java service1CoordinatorImpl.java service1Impl2.java service1CompositeImpl.java
Expected output	Positive test: "JCA_2008 request serviceComposite operation1 invoked service1 operation1 invoked serviceComposite operation1 Initial service2 operation1 invoked serviceComposite operation1 1 service3 operation1 invoked serviceComposite operation1 2 service4 operation1 invoked serviceComposite operation1 3"

149

150 JCA_2009_TestCase

151

Testcase ID	JCA_2009_TestCase
Test Assertion	JCA-TA-2011, JCA-TA-10004
Description	Tests that where one component is a client of a service provided by a second component, both with Java implementations and which both run in the same JVM, and the client reference is marked with @AllowsPassBy-Reference but the service implementation methods are not marked with @AllowsPassByReference that invocations of the service use "pass by value" semantics.
Artifacts	JCA_2009_TestCase.java Test_JCA_2009.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service4.java service1Impl7.java service4Impl.java
Expected output	Positive test: "JCA_2009 request service1 operation1 invoked service2 operation1 in-voked request+1"

152

153 JCA_2010_TestCase

154

Testcase ID	JCA_2010_TestCase
Test Assertion	JCA-TA-2011, JCA-TA-10004
Description	Tests that where one component is a client of a service provided by a second component, both with Java implementations and which both run in the same JVM, and the service implementation methods are not marked with @AllowsPassByReference but the client reference is not marked with @AllowsPassByReference that invocations of the service use "pass by value" semantics.
Artifacts	JCA_2010_TestCase.java Test_JCA_2010.composite TestInvocation.wsdl TestClient_0002.composite

	Service1.java Service4.java service1Impl7b.java service4Impl1.java
Expected output	Positive test: "JCA_2010 request service1 operation1 invoked service2 operation1 invoked request+1"

2.3 Section 3

JCA_3001_TestCase

Testcase ID	JCA_3001_TestCase
Test Assertion	JCA-TA-3001
Description	Tests that a Java Interface class name is fully qualified on a service <interface.java> element
Artifacts	JCA_3001_TestCase.java Test_JCA_3001.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1Impl.java
Expected output	Negative test: “exception”

JCA_3002_TestCase

Testcase ID	JCA_3002_TestCase
Test Assertion	JCA-TA-3001
Description	Tests that a Java Interface is fully qualified on a reference <interface.java> element
Artifacts	JCA_3002_TestCase.java

	Test_JCA_3002.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1Impl.java Service1Impl2.java
Expected output	Negative test: “exception”

JCA_3003_TestCase

Testcase ID	JCA_3003_TestCase
Test Assertion	JCA-TA-3002, JCA-TA-10031
Description	Tests that callback interfaces on a service are specified in their fully qualified form
Artifacts	JCA_3003_TestCase.java Test_JCA_3003.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1CalbackImpl.java Service3WithCallback.java Service3Callback.java service3Impl1.java
Expected output	Negative test: “exception”

JCA_3004_TestCase

Testcase ID	JCA_3004_TestCase

Test Assertion	JCA-TA-3002, JCA-TA-10031
Description	Tests that callback interfaces on a reference are specified in their fully qualified form
Artifacts	JCA_3004_TestCase.java Test_JCA_3004.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1CalbackImpl.java Service3WithCallback.java Service3Callback.java service3Impl1.java
Expected output	Negative test: “exception”

169

170 ## JCA_3005_TestCase

171

Testcase ID	JCA_3005_TestCase
Test Assertion	JCA-TA-3003, JCA-TA-10031, JCA-TA-10064
Description	Tests that callback interfaces specified in the composite match the callback interface specified in the service interface class
Artifacts	JCA_3005_TestCase.java Test_JCA_3005.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1CalbackImpl.java Service3WithCallback.java Service3Callback.java service3Impl1.java
Expected output	Positive test: "JCA_3005 request service1 operation1 invoked service3 operation1 in-

	voked service1 callback1 invoked"
~~Testcase ID~~	~~JCA_3005_TestCase~~
~~Test Assertion~~	~~JCA-TA-3003, JCA-TA-10031~~
~~Description~~	~~Tests that callback interfaces specified in the composite match the callback interface specified in the service interface class~~
~~Artifacts~~	~~JCA_3005_TestCase.java~~ ~~Test_JCA_3005.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~ASM_0002_Client.java~~ ~~Service1.java~~ ~~service1CalbackImpl.java~~ ~~Service3WithCallback.java~~ ~~Service3Callback.java~~ ~~service3Impl1.java~~
~~Expected output~~	~~Positive test:~~ ~~"JCA_3005 request service1 operation1 invoked service3 operation1 invoked service1 callback1 invoked"~~

JCA_3006_TestCase

Testcase ID	JCA_3006_TestCase
Test Assertion	JCA-TA-3003, JCA-TA-10031
Description	Tests that callback interfaces specified in the composite match the callback interface specified in the service interface class
Artifacts	JCA_3006_TestCase.java Test_JCA_3006.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1CalbackImpl.java Service3WithCallback.java Service3Callback.java service3Impl1.java
Expected output	Negative test: "exception"

JCA_3007_TestCase

Testcase ID	JCA_3007_TestCase
Test Assertion	JCA-TA-3003, JCA-TA-10031
Description	Tests that callback interfaces specified in the composite match the callback interface specified in the reference interface class
Artifacts	JCA_3007_TestCase.java Test_JCA_3007.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1CalbackImpl.java Service3WithCallback.java Service3Callback.java service3Impl1.java
Expected output	Negative test: "exception"

JCA_3008_TestCase

Testcase ID	JCA_3008_TestCase
Test Assertion	JCA-TA-3004
Description	Tests that <interface.java/> conforms to the schema
Artifacts	JCA_3008_TestCase.java Test_JCA_3008.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java
Expected output	Negative test: "exception"

JCA_3009_TestCase

Testcase ID	JCA_3009_TestCase
Test Assertion	JCA-TA-3005
Description	Tests that remotable attribute matches @Remotable annotation
Artifacts	JCA_3009_TestCase.java Test_JCA_3009.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java
Expected output	Negative test: "exception"

JCA_3010_TestCase

Testcase ID	JCA_3010_TestCase
Test Assertion	JCA-TA-3005
Description	Tests that remotable attribute matches @Remotable annotation
Artifacts	JCA_3010_TestCase.java Test_JCA_3010.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java
Expected output	Positive test: "JCA_3010 request service1 operation1 invoked"

JCA_3011_TestCase

Testcase ID	JCA_3011_TestCase
Test Assertion	JCA-TA-3006
Description	Tests that a service interfaces doesn't contain forbidden annotations
Artifacts	JCA_3011_TestCase.java Test_JCA_3011.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java JCA3011Service.java JCA3011serviceImpl.java
Expected output	Negative test: "exception"

JCA_3012_TestCase

Testcase ID	JCA_3012_TestCase
Test Assertion	JCA-TA-3007
Description	Tests that callback interfaces don't contain forbidden annotations
Artifacts	JCA_3012_TestCase.java Test_JCA_3012.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service3WithCallback.java Service1.java service1CallbackImpl.java Service3Callback.java JCA3012Service3WithCallback.java JCA3012Service3Callback.java JCA3012service3Impl1.java
Expected output	Negative test:

	"exception"

JCA_3013_TestCase

Testcase ID	JCA_3013_TestCase
Test Assertion	JCA-TA-3008
Description	Tests that where two Java interfaces are otherwise compatible, every method present in both interfaces where annotated with @OneWay in one interface is also annotated with @OneWay in the other interface
Artifacts	JCA_3013_TestCase.java Test_JCA_3013.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java ServiceOneWayMissing.java serviceOneWayImpl.java Service1OneWayMissingCallerImpl.java ServiceOneWay.java
Expected output	Negative test: "exception"

JCA_3014_TestCase

Testcase ID	JCA_3014_TestCase
Test Assertion	JCA-TA-3009
Description	Tests that where an <interface.java/> element references a Java interface class which contains a @WebService annotation with a non-empty wsdlLocation property pointing at a WSDL document, it is treated as if it were an <interface.wsdl/> element with an @interface attribute identifying the portType in the WSDL document mapped from the Java interface class
Artifacts	JCA_3014_TestCase.java Test_JCA_3014.composite TestInvocation.wsdl TestClient_0002.composite

	ASM_0002_Client.java Service1.java Service3Operations.java Service3OperationsWSDL.java Service3OperationsWSDLImpl.java Service1Calls3OperationsImpl.java Service3OperationsWSDL.wsdl
Expected output	Negative test: "exception"

199

200

201 ## 2.4 Section 4

202 ## JCA_4001_TestCase

203

Testcase ID	JCA_4001_TestCase
Test Assertion	JCA-TA-4001, JCA-TA-10005, JCA-TA-10006
Description	Tests that for a stateless Java implementation, that all lifecycle stages are performed in the correct sequence and that no stage occurs out of sequence.
Artifacts	JCA_4001_TestCase.java Test_JCA_4001.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1StatelessLifecycleImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4001 request Constructing property1 injected reference1 injected property2 injected reference2 injected Init invoked Init completed serviceLifecycle operation1 invoked service1 operation1 invoked service2 operation1 invoked service3 operation1 invoked prop1 prop2 prop3 Destroy invoked"

204

205 ## JCA_4002_TestCase

206

Testcase ID	JCA_4002_TestCase
Test Assertion	JCA-TA-4004
Description	Tests that where a stateless Java implementation throws an exception from its Constructor method, that the implementation transitions to the Terminating state
Artifacts	JCA_4002_TestCase.java Test_JCA_4002.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1LifecycleExceptionsImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4002 request Constructing exception thrown"

207

208 ## JCA_4003_TestCase

209

Testcase ID	JCA_4003_TestCase
Test Assertion	JCA-TA-4010
Description	Tests that when a stateless Java implementation throws an exception when a property is injected, that the implementation transitions to the Destroying state
Artifacts	JCA_4003_TestCase.java Test_JCA_4003.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java

	service1LifecycleExceptionsImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4003 request Constructing property1 injected reference1 injected property2 injected exception thrown Destroy invoked"

JCA_4004_TestCase

Testcase ID	JCA_4004_TestCase
Test Assertion	JCA-TA-4012
Description	Tests that where a Java implementation invokes an injected reference while in the initializing phase and the target of the reference has not been initialized, that the implementation receives a ServiceUnavailableException
Artifacts	JCA_4004_TestCase.java Test_JCA_4004.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1LifecycleExceptionsImpl.java service1UninitImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4004 request Constructing property1 injected reference1 injected property2 injected reference2 injected Init invoked calling uninitialized service ServiceUnavailable exception received Init completed serviceLifecycle operation1 invoked service1 operation1 invoked service2 operation1 invoked service3 operation1 invoked prop1 prop2 prop3 Destroy invoked"

JCA_4005_TestCase

Testcase ID	JCA_4005_TestCase
Test Assertion	JCA-TA-4015
Description	Tests that where a Java implementation throws an exception from the

	method marked with the @Init annotation, that the implementation transitions to the Destroying state
Artifacts	JCA_4005_TestCase.java Test_JCA_4005.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1LifecycleExceptionsImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4005 request Constructing property1 injected reference1 injected property2 injected reference2 injected Init invoked exception thrown Destroy invoked"

216

217 ## JCA_4006_TestCase

218

Testcase ID	JCA_4006_TestCase
Test Assertion	JCA-TA-4019
Description	Tests that a Java implementation in the Destroying state which invokes a method on a reference whose target service has already been destroyed receives an InvalidServiceException
Artifacts	JCA_4006_TestCase.java Test_JCA_4006.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1StatelessLifecycleImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4006 request Constructing property1 injected reference1 injected

	property2 injected reference2 injected Init invoked Init completed serviceLifecycle operation1 invoked service1 operation1 invoked service2 operation1 invoked service3 operation1 invoked prop1 prop2 prop3 Destroy invoked"

219

220 ## JCA_4007_TestCase

221

Testcase ID	JCA_4007_TestCase
Test Assertion	JCA-TA-4022
Description	Tests that where a Java implementation throws an exception from its method marked with the @Destroy annotation, that the implementation transitions to the terminated state
Artifacts	JCA_4007_TestCase.java Test_JCA_4007.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1LifecycleExceptionsImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4007 request Constructing property1 injected reference1 injected property2 injected reference2 injected Init invoked Init completed serviceLifecycle operation1 invoked service1 operation1 invoked service2 operation1 invoked service3 operation1 invoked prop1 prop2 prop3 Destroy invoked exception thrown"

222

223 ## JCA_4008_TestCase

224

Testcase ID	JCA_4008_TestCase
Test Assertion	JCA-TA-4024
Description	Tests that when a stateless Java implementation throws an exception when a reference is injected, that the implementation transitions to the Destroying state
Artifacts	JCA_4008_TestCase.java

	Test_JCA_4008.composite TestInvocation.wsdl TestClient_0002.composite JCA_0002_Client.java Service1.wsdl DataStore.java lifecycleControllerImpl.java service1LifecycleExceptionsImpl.java service1Impl.java dataStoreCompositeImpl.java
Expected output	"JCA_4008 request Constructing property1 injected reference1 injected property2 injected reference2 injected exception thrown Destroy invoked"

225

226 2.5 Section 5

227 JCA_7001_TestCase

228

Testcase ID	JCA_7001_TestCase
Test Assertion	JCA-TA-7001 JCA-TA-7002
Description	Tests that where a Java implementation offers a bidirectional service and has fields and setter methods annotated with @Callback, when the bidirectional service is invoked, a callback reference object is injected into those fields and setter methods which have a type which is the interface for the callback service but null is injected into fields and setter methods which have a different type.
Artifacts	JCA_7001_TestCase.java Test_JCA_7001.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3WithCallback.java Service3Callback.java Service7WithCallback.java Service7Callback.java service1CallbackImpl.java MultipleCallbacksImpl.java

Expected output	"JCA_7001 request service1 operation1 invoked service2 operation1 invoked callback1 invoked other callbacks null";

JCA_7002_TestCase

Testcase ID	JCA_7002_TestCase
Test Assertion	JCA-TA-7002
Description	Tests that where a Java implementation offers a bidirectional service and also has one non-bidirectional service and has fields and setter methods annotated with @Callback, when the non-bidirectional service is invoked, null is injected into all the fields and setter methods annotated with @Callback
Artifacts	JCA_7002_TestCase.java Test_JCA_7002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3WithCallback.java Service3Callback.java Service7WithCallback.java Service7Callback.java service1CallbackImpl.java MultipleCallbacksImpl.java
Expected output	"JCA_7002 request service1 operation1 invoked service2 operation1 invoked callback1 invoked other callbacks null";

JCA_7003_TestCase

Testcase ID	JCA_7003_TestCase
Test Assertion	JCA-TA-7003 JCA-TA-7004 JCA-TA-7005 JCA-TA-7007
Description	Tests that where a Java interface class is an asynchronous service mapping of a WSDL portType containing request/response operations that the class is annotated with the "asyncInvocation" intent
Artifacts	JCA_7003_TestCase.java

	Test_JCA_7003.composite TestInvocation.wsdl TestClient_0002.composite Service1AsyncServer.java Service1AsyncServerImpl.java
Expected output	"JCA_7003 request service1 operation1 invoked asynchronously"

235

236 JCA_7004_TestCase

237

Testcase ID	JCA_7004_TestCase
Test Assertion	JCA-TA-7003
Description	Tests that where a Java interface class is an asynchronous service mapping of a WSDL portType containing request/response operations that the class is annotated with the "asyncInvocation" intent - this is the negative version of JCA_7003_Testcase where the asyncInvocation intent is left off the interface class
Artifacts	JCA_7004_TestCase.java Test_JCA_7004.composite TestInvocation.wsdl TestClient_0002.composite Service1AsyncServerError.java Service1AsyncServerErrorImpl.java
Expected output	"exception"

238

239 JCA_7005_TestCase

240

Testcase ID	JCA_7005_TestCase
Test Assertion	JCA-TA-7006
Description	Tests that when an asynchronous service implementation invokes the sendResponse method of the supplied ResponseDispatch.object when that method has already been called once, the method throws an IllegalStateException
Artifacts	JCA_7005_TestCase.java Test_JCA_7005.composite TestInvocation.wsdl TestClient_0002.composite

	Service1AsyncServer.java Service1AsyncServerMultiReponseImpl.java asyncControllerImpl.java dataStoreCompositeImpl.java
Expected output	"JCA_7005 request service1 operation1 invoked asynchronously IllegalStateException thrown on second invocation of ResponseDispatch.sendResponse() method"

241

242 JCA_7006_TestCase

243

Testcase ID	JCA_7006_TestCase
Test Assertion	JCA-TA-9068 JCA-TA-9069
Description	Tests that when an asynchronous service implementation invokes the sendFault method of the supplied ResponseDispatch.object that the exception is sent to the client of the service operation and if the sendFault method is subsequently invoked a second time when that method has already been called once for a given ResponseDispatch object, the method throws an IllegalStateException
Artifacts	JCA_7006_TestCase.java Test_JCA_7006.composite TestInvocation.wsdl TestClient_0002.composite Service1AsyncServer.java Service1AsyncServerMultiFaultImpl.java asyncControllerImpl.java dataStoreCompositeImpl.java
Expected output	"JCA_7005 request service1 operation1 invoked asynchronously IllegalStateException thrown on second invocation of ResponseDispatch.sendResponse() method"

244
245

246

247 Section 8

248 JCA_8001_TestCase

249

Testcase ID	JCA_8001_TestCase

Test Assertion	JCA-TA-8001
Description	Tests that intent annotations use the @Intent annotation in the definition of the annotation. In this testcase, two mutually exclusive intents are defined in the domain. One of them is also defined as a Java annotation using the @Intent definition. TEST_JCA_8001Component1 requires both mutually exclusive intents (one in the SCDL, one in the Java implementation class) and should flag an error that mutually exclusive intents are not allowed.
Artifacts	JCA_8001_TestCase.java Test_JCA_8001.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1Intent.java TestIntent2.java definitions.xml
Expected output	Negative test: "exception"

250

251 ## JCA_8002_TestCase

252

Testcase ID	JCA_8002_TestCase
Test Assertion	JCA-TA-8002
Description	Tests that intent annotations cannot be used on methods which are not reference setter methods.
Artifacts	JCA_8002_TestCase.java Test_JCA_8002.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java TestIntent2.java service1BadIntent.java
Expected output	Negative test: "exception"

253

JCA_8003_TestCase

Testcase ID	JCA_8003_TestCase
Test Assertion	JCA-TA-8003
Description	Tests that intent annotations cannot be used on fields which are not references.
Artifacts	JCA_8003_TestCase.java Test_JCA_8003.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java TestIntent2.java service1BadIntent.java
Expected output	Negative test: "exception"

JCA_8004_TestCase

Testcase ID	JCA_8004_TestCase
Test Assertion	JCA-TA-8004
Description	Tests that intent annotations cannot be used on constructor parameters which are not references.
Artifacts	JCA_8004_TestCase.java Test_JCA_8004.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java TestIntent2.java service1BadIntent.java
Expected output	Negative test: "exception"

JCA_8005_TestCase

Testcase ID	JCA_8005_TestCase
Test Assertion	JCA-TA-8002, JCA-TA-8003, JCA-TA-8004
Description	Tests that intent annotations can be used to annotate every place that a reference is allowed (setter method, field, or constructor parameter).
Artifacts	JCA_8005_TestCase.java Test_JCA_8005.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java TestIntent2.java service1GoodIntent.java
Expected output	Positive test: "JCA_8005 request service1 operation1 invoked service2 operation1 invoked service3 operation1 invoked service4 operation1 invoked"

JCA_8006_TestCase

Testcase ID	JCA_8006_TestCase
Test Assertion	JCA-TA-8005
Description	Tests that intent annotations are merged together (unioned) when present on the same Java element. In this testcase, two mutually exclusive intents are placed on the same Java element. When the runtime correctly merges them, an error should result.
Artifacts	JCA_8006_TestCase.java Test_JCA_8006.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java TestIntent1.java TestIntent2.java service1BadIntent.java definitions.xml

Expected output	Negative test: "exception"

265

266 JCA_8007_TestCase

267

Testcase ID	JCA_8007_TestCase
Test Assertion	JCA-TA-8006
Description	Tests that intent annotations are correctly merged when they appear at different levels in a Java interface class. PolicySet1 satisfies testIntent3 which is the only intent that should be applicable to the service after the intents in Service5Intents are normalized.
Artifacts	JCA_8007_TestCase.java Test_JCA_8007.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java TestIntent3.java TestIntent4.java service5Impl2.java service5Impl.java Service5Intents.java definitions.xml
Expected output	Positive test: "JCA_8007 request service1 operation1 invoked service2 operation1 invoked"

268

269 JCA_8008_TestCase

270

Testcase ID	JCA_8008_TestCase
Test Assertion	JCA-TA-8007
Description	Tests that policySets cannot be used on methods which are not reference setter methods.
Artifacts	JCA_8008_TestCase.java Test_JCA_8008.composite

	TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1BadPolicySet.java definitions.xml
Expected output	Negative test: "exception"

271

272 ## JCA_8009_TestCase

273

Testcase ID	JCA_8009_TestCase
Test Assertion	JCA-TA-8008
Description	Tests that policySets cannot be used on fields which are not references.
Artifacts	JCA_8009_TestCase.java Test_JCA_8009.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1BadPolicySet.java definitions.xml
Expected output	Negative test: "exception"

274

275

Testcase ID	JCA_8010_TestCase
Test Assertion	JCA-TA-8009
Description	Tests that policySets cannot be used on constructor parameters which are not references.
Artifacts	JCA_8010_TestCase.java Test_JCA_8010.composite TestInvocation.wsdl

	TestClient_0002.composite ASM_0002_Client.java Service1.java service1BadPolicySet.java definitions.xml
Expected output	Negative test: "exception"

276

277 ## JCA_8011_TestCase

278

Testcase ID	JCA_8011_TestCase
Test Assertion	JCA-TA-8010
Description	Tests that policySet annotations are correctly merged when they appear at different levels in a Java interface class. PolicySet1 satisfies testIntent3 and PolicySet2 satisfies testIntent5.
Artifacts	JCA_8010_TestCase.java Test_JCA_8010.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service6Impl.java service6Impl2.java Service6PolicySets.java TestIntent3.java TestIntent4.java definitions.xml
Expected output	Positive test: "JCA_8011 request service1 operation1 invoked service2 operation1 invoked"

279

280

2.6 Section 9

JCA_9001_TestCase

Testcase ID	JCA_9001_TestCase
Test Assertion	JCA-TA-9001
Description	Tests that if an invocation of the getService() method of the Component-Context API is made for a reference that is 0..n or 1..n multiplicity, that the caller receives an IllegalArgumentException
Artifacts	JCA_9001_TestCase.java Test_JCA_9001.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1ContextImpl1.java service1Impl.java
Expected output	Positive test: "JCA_9001 request invokerService operation1 invoked IllegalArgumentException received"

JCA_9002_TestCase

Testcase ID	JCA_9002_TestCase
Test Assertion	JCA-TA-9002
Description	Tests that if an invocation of the getRequestContext() method of the ComponentContext API is made during the execution of a Java business method of a service operation on the same Java thread used to invoke the business method, then a non-null value is returned for the RequestContext
Artifacts	JCA_9002_TestCase.java Test_JCA_9002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1ContextImpl1.java service1Impl.java
Expected output	Positive test:

	"JCA_9002 request service1 operation1 invoked RequestContext - serviceName: Service1"

287

288 JCA_9003_TestCase

289

Testcase ID	JCA_9003_TestCase
Test Assertion	JCA-TA-9003
Description	Tests that if an invocation of the getServiceReference() method of the RequestContext API is made during the execution of a Java business method, then the method returns a ServiceReference object that represents the service that was invoked
Artifacts	JCA_9003_TestCase.java Test_JCA_9003.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1Superset.java service1RequestContextImpl2.java
Expected output	Positive test: "JCA_9003 request service1 operation1 invoked ServiceReference obtained: service1 checkService operation2 invoked"

290

291 JCA_9004_TestCase

292

Testcase ID	JCA_9004_TestCase
Test Assertion	JCA-TA-9004
Description	Tests that if an invocation of the getServiceReference() method of the RequestContext API is made during the execution of a Java business method of a callback, then the method returns a ServiceReference object that represents the callback that was invoked
Artifacts	JCA_9004_TestCase.java Test_JCA_9004.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3WithCallback.java

	Service3Callback.java service1CallbackContextImpl1.java service3Impl1.java
Expected output	Positive test: "JCA_9004 request serviceComposite1 operation1 invoked callbackService operation1 invokedserviceComposite1 callback1 invoked ServiceReference obtained: serviceComposite1 check callback1 invoked"

293

294 ## JCA_9005_TestCase

295

Testcase ID	JCA_9005_TestCase
Test Assertion	JCA-TA-9005
Description	Tests that if an invocation of the getRequestContext() method of the ComponentContext API is made during the execution of a Java business method of a callback operation on the same Java thread used to invoke the business method, then a non-null value is returned for the RequestContext
Artifacts	JCA_9005_TestCase.java Test_JCA_9005.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3WithCallback.java Service3Callback.java service1CallbackContextImpl2.java service3Impl1.java
Expected output	Positive test: "JCA_9005 request serviceComposite1 operation1 invoked callbackService operation1 invoked serviceComposite1 callback1 invoked RequestContext - serviceName: reference1"

296

297 ## JCA_9006_TestCase

298

Testcase ID	JCA_9006_TestCase
Test Assertion	JCA-TA-9070 JCA-TA-9006 JCA-TA-9007 JCA-TA-9008

	JCA-TA-9014
Description	Tests invocations of the ComponentContext.getServiceReference() method o invoked with a referenceName parameter for a reference of multiplicity 0..n o invoked for a reference which does not have the interface of the type in the businessInterface parameter o invoked with a referenceName value that does not match the name of any of the references of the component o invoked for a reference with multiplicity 0..1 which has no target service configured o invoked for a valid reference with a target service configured
Artifacts	JCA_9006_TestCase.java Test_JCA_9006.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1Impl.java Service1CCgetServiceReferenceImpl.java
Expected output	"JCA_9006 request service1 operation1 invoked getServiceReference expected IllegalArgumentException received for 0..n ref expected IllegalArgumentException received for invalid businessInterface expected IllegalArgumentException received for invalid reference name expected null ServiceReference for unwired 0..1 reference expected non-null ServiceReference for wired 1..1 reference service2 operation1 invoked" ;

299

300 ## JCA_9007_TestCase

301

Testcase ID	JCA_9007_TestCase
Test Assertion	JCA-TA-9009
Description	Tests invocations of the ComponentContext.getURI() method: o invocation returns a String containing the absolute URI of the component in the SCA Domain
Artifacts	JCA_9007_TestCase.java Test_JCA_9007.composite TestInvocation.wsdl TestClient_0002.composite Service1.java

	Service1CCgetURIImpl.java
Expected output	"JCA_9007 request service1 operation1 invoked getURI getURI() returned URI = TEST_JCA_9007Component1"

302

303 ## JCA_9008_TestCase

304

Testcase ID	JCA_8_TestCase
Test Assertion	JCA-TA-9001 JCA-TA-9010 JCA-TA-9011 JCA-TA-9012 JCA-TA-9013
Description	Tests invocations of the ComponentContext.getService() method: o method throws an IllegalArgumentException if the referenceName reference has multiplicity 0..n o method returns a proxy object with the businessInterface for the reference referenceName, where that reference has a target service configured o method returns null for a 0..1 reference with no target service configured o method throws an IllegalArgumentException where the component does not have a reference with the name referenceName o method throws an IllegalArgumentException where the the referenceName reference does not have the interface defined in the businessInterface parameter
Artifacts	JCA_9008_TestCase.java Test_JCA_9008.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1CCgetServiceImpl.java
Expected output	"JCA_9008 request service1 operation1 invoked getService expected IllegalArgumentException received for 0..n ref expected IllegalArgumentException received for invalid businessInterface expected IllegalArgumentException received for invalid reference name expected null Service for unwired 0..1 reference expected non-null Service for wired 1..1 reference service2 operation1 invoked"

305

306 ## JCA_9009_TestCase

307

Testcase ID	JCA_9009_TestCase

Test Assertion	JCA-TA-9015 JCA-TA-9016 JCA-TA-9017 JCA-TA-9018 JCA-TA-9019
Description	Tests invocations of the ComponentContext.getServices() method: o invocation returns a Collection containing a proxy object for each target service of the reference each implementing the interface supplied in the businessInterface parameter o invocation returns an empty Collection where the reference named by referenceName exists but has no target services configured o invocation throws an IllegalArgumentExeption when the named reference exists and has a multiplicity of 0..1 or 1..1 o invocation throws an IllegalArgumentExeption when the named reference does not exist o invocation throws an IllegalArgumentExeption when the named reference does not have an interface compatble with the interface in the businessInterface parameter
Artifacts	JCA_9009_TestCase.java Test_JCA_9009.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1CCgetServicesImpl.java
Expected output	"JCA_9009 request service1 operation1 invoked getServices expected IllegalArgumentException received for 1..1 ref expected IllegalArgumentException received for invalid businessInterface expected IllegalArgumentException received for invalid reference name expected empty Collection for unwired 0..n reference expected non-empty Collection for wired 0..n reference service2 operation1 invoked service3 operation1 invoked"

308

309 ## JCA_9010_TestCase

310

Testcase ID	JCA_9010_TestCase
Test Assertion	JCA-TA-9020 JCA-TA-9021 JCA-TA-9022 JCA-TA-9023 JCA-TA-9024
Description	Tests invocations of the ComponentContext.getServiceReferences() method: o invocation returns a Collection containing a ServiceReference object for each target service of the reference each implementing the interface sup-

	plied in the businessInterface parameter o invocation returns an empty Collection where the reference named by referenceName exists but has no target services configured o invocation throws an IllegalArgumentExeption when the named reference exists and has a multiplicity of 0..1 or 1..1 o invocation throws an IllegalArgumentExeption when the named reference does not exist o invocation throws an IllegalArgumentExeption when the named reference does not have an interface compatble with the interface in the businessInterface parameter
Artifacts	JCA_9010_TestCase.java Test_JCA_9010.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1CCgetServiceReferencesImpl.java
Expected output	"JCA_9010 request service1 operation1 invoked getServiceReferences expected IllegalArgumentException received for 1..1 ref expected IllegalArgumentException received for invalid businessInterface expected IllegalArgumentException received for invalid reference name expected empty Collection for unwired 0..n reference expected non-empty Collection for wired 0..n reference service2 operation1 invoked service3 operation1 invoked"

311

312 ## JCA_9011_TestCase

313

Testcase ID	JCA_9011_TestCase
Test Assertion	JCA-TA-9025 JCA-TA-9026
Description	Tests invocations of the ComponentContext.createSelfReference(businessInterface) method: o invocation returns a ServiceReference typed by the interface of the businessInterface parameter for one of the services offered by the component that has that interface o method thorws an IllegalArgumentException if the component has no service which has the interface specified by the businessInterface parameter
Artifacts	JCA_9011_TestCase.java Test_JCA_9011.composite TestInvocation.wsdl TestClient_0002.composite

	Service1.java Service1CCcreateSelfReferenceAImpl.java
Expected output	"JCA_9011 request service1 operation1 invoked createSelfReference expected IllegalArgumentException received for invalid businessInterface" expected non-null ServiceReference service1 operation2 invoked"

314

315 JCA_9012_TestCase

316

Testcase ID	JCA_9012_TestCase
Test Assertion	JCA-TA-9027 JCA-TA-9028 JCA-TA-9029
Description	Tests invocations of the ComponentContext.createSelfReference(businessInterface, serviceName) method: o invocation returns a ServiceReference typed by the interface of the businessInterface parameter for the service offered by the component that has that interface and has the name specified by serviceName o method throws an IllegalArgumentException if the service named in the serviceName parameter does not have the interface specified by the businessInterface parameter o method throws an IllegalArgumentException if the component does not have a service with the name specified in the serviceName parameter
Artifacts	JCA_9012_TestCase.java Test_JCA_9012.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1CCcreateSelfReferenceBImpl.java
Expected output	"JCA_9012 request service1 operation1 invoked createSelfReference expected IllegalArgumentException received for invalid businessInterface expected IllegalArgumentException received for invalid service name expected non-null ServiceReference service1 operation2 invoked"

317

318 JCA_9013_TestCase

319

Testcase ID	JCA_9013_TestCase
Test Assertion	JCA-TA-9030 JCA-TA-9031 JCA-TA-9032

	JCA-TA-9033
Description	Tests invocations of the ComponentContext.getProperty method: o invocation returns an object of the type specified by the type parameter containing the value in the component configuration for the property named by the propertyName parameter o method returns null for a property which has no value specified in the component configuration o method throws an IllegalArgumentException if the component does not have a property with the name specified in the propertyName parameter o method throws an IllegalArgumentException if the property named in the propertyName parameter does not have a type compatible with the supplied type parameter
Artifacts	JCA_9013_TestCase.java Test_JCA_9013.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1CCgetPropertyImpl.java
Expected output	"JCA_9013 request service1 operation1 invoked getProperty expected IllegalArgumentException received for invalid property type" expected IllegalArgumentException received for invalid property name expected null value for unconfigured optional property expected non-null property value for required property: 2.456"

320

321 ## JCA_9014_TestCase

322

Testcase ID	JCA_9014_TestCase
Test Assertion	JCA-TA-9034 JCA-TA-9035
Description	Tests invocations of the ComponentContext.cast method: o invocation returns a ServiceReference object typed by the same interface as specified by the reference proxy object supplied in the target parameter o method throws an IllegalArgumentException if the supplied target parameter is not an SCA reference proxy object
Artifacts	JCA_9014_TestCase.java Test_JCA_9014.composite TestInvocation.wsdl TestClient_0002.composite

	Service1.java Service1CCcastImpl.java
Expected output	"JCA_9014 request service1 operation1 invoked cast expected IllegalArgumentException received for invalid reference proxy expected non-null ServiceReference ServiceReference has correct business interface"

323

324 JCA_9015_TestCase

325

Testcase ID	JCA_5_TestCase
Test Assertion	JCA-TA-9039 JCA-TA-9040 JCA-TA-9041 JCA-TA-9042 JCA-TA-9043 JCA-TA-9044 JCA-TA-9045 JCA-TA-9046 JCA-TA-9047 JCA-TA-9048
Description	Tests invocations of the RequestContext.method: o getServiceName invocation returns the name of the service for which the operation is being processed when called from a thread processing a service operation o getServiceName method returns null when called from a thread not processing a service operation or callback operation o getCallbackReference returns a ServiceReference typed by the interface of the callback when invoked by a thread processing an operation of a bidirectional service o getCallbackReference returns null when invoked by a thread processing an operation of a service which is not bidirectional o getCallbackReference returns null when invoked by a thread not processing an operation of a service o getCallback returns a reference proxy typed by the interface of the callback when invoked by a thread processing an operation of a bidirectional service o getCallback returns null when invoked by a thread processing an operation of a service which is not bidirectional o getCallback returns null when invoked by a thread not processing an operation of a service o getServiceReference method returns a ServiceReference representing the callback service when invoked from a thread processing a callback operation o getServiceReference method returns null when called from a thread not involved in processing either a service operation or a callback operation

Artifacts	JCA_9015_TestCase.java Test_JCA_9015.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1RCgetServiceNameImpl.java
Expected output	"JCA_9015 request service1 operation1 invoked RequestContext expected non-null serviceName from getServiceName expected value for returned serviceName expected null ServiceReference from getCallbackReference expected null reference proxy from getCallbackservice2 operation1 invoked expected non-null ServiceReference from getCallbackReference expected ServiceReference has the callback business interfaceservice2 callback1 invoked expected non-null reference proxy from getCallback expected reference proxy has the callback business interfaceservice2 callback1 invokedservice3 operation1 invoked expected non-null ServiceReference from getServiceReference expected ServiceReference has the callback business interface expected null from getServiceName" expected null ServiceReference from getCallbackReference expected null reference proxy from getCallback expected null ServiceReference from getServiceReference"
~~Testcase ID~~	~~JCA_5_TestCase~~
~~Test Assertion~~	~~JCA-TA-9039~~ ~~JCA-TA-9040~~ ~~JCA-TA-9041~~ ~~JCA-TA-9042~~ ~~JCA-TA-9043~~ ~~JCA-TA-9044~~ ~~JCA-TA-9045~~ ~~JCA-TA-9046~~ ~~JCA-TA-9047~~ ~~JCA-TA-9048~~
~~Description~~	~~Tests invocations of the RequestContext.method:~~ ~~o getServiceName invocation returns the name of the service for which the operation is being processed when called from a thread processing a service operation~~ ~~o getServiceName method returns null when called from a thread not processing a service operation or callback operation~~ ~~o getCallbackReference returns a ServiceReference typed by the interface of the callback when invoked by a thread processing an operation of a bidirectional service~~ ~~o getCallbackReference returns null when invoked by a thread processing an operation of a service which is not bidirectional~~ ~~o getCallbackReference returns null when invoked by a thread not processing an operation of a service~~ ~~o getCallback returns a reference proxy typed by the interface of the callback when invoked by a thread processing an operation of a bidirectional service~~ ~~o getCallback returns null when invoked by a thread processing an operation of a service which is not bidirectional~~ ~~o getCallback returns null when invoked by a thread not processing an operation of a service~~ ~~o getServiceReference method returns a ServiceReference representing the callback service when invoked from a thread processing a callback op-~~

	~~eration~~ ~~o getServiceReference method returns null when called from a thread not involved in processing either a service operation or a callback operation~~
~~Artifacts~~	~~JCA_9015_TestCase.java~~ ~~Test_JCA_9015.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.java~~ ~~Service1RCgetServiceNameImpl.java~~
~~Expected output~~	

326

327 ## JCA_9016_TestCase

328

Testcase ID	JCA_9016_TestCase
Test Assertion	JCA-TA-9051 JCA-TA-9052 JCA-TA-9053 JCA-TA-9054 JCA-TA-9055 JCA-TA-9056 JCA-TA-9057 JCA-TA-9058 JCA-TA-9059 JCA-TA-9060
Description	Tests invocations of the SCAClientFactory API class: o invocation of newInstance(URI) with a valid URI value returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter o invocation of newInstance(URI) with an invalid URI value causes a NoSuchDomainException to be thrown o invocation of newInstance(Properties, URI) with a valid URI value returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter o invocation of newInstance(Properties, URI) with an invalid URI value causes a NoSuchDomainException to be thrown o invocation of newInstance(Classloader, URI) with a valid URI value returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter o invocation of newInstance(Classloader, URI) with an invalid URI value causes a NoSuchDomainException to be thrown o invocation of newInstance(Properties, Classloader, URI) with a valid URI value returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter o invocation of newInstance(Properties, Classloader, URI) with an invalid URI value causes a NoSuchDomainException to be thrown o invocation of getService with a serviceURI that identifies a service in the Domain which has the business interface identified bby the interfaze para-

	meter returns a reference proxy object which can be used to invoke operations on the identified service o invocation of getService with a serviceURI for which there is no service in the domain throws a NoSuchServiceException
Artifacts	JCA_9016_TestCase.java Test_JCA_9016.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1SCAClientImpl.java
Expected output	"JCA_9016 request service1 operation1 invoked SCAClient expected non-null client factory expected exception received for invalid URI: NoSuchDomainException expected non-null client factory expected exception received for invalid URI: NoSuchDomainException expected non-null client factory expected exception received for invalid URI: NoSuchDomainExceptionexpected non-null client factory expected exception received for invalid URI: NoSuchDomainException expected non-null reference proxyservice2 operation1 invoked expected NoSuchServiceException exception received for getService()"

2.7 Section 10

JCA_10001_TestCase

Testcase ID	JCA_10001_TestCase
Test Assertion	JCA-TA-10001
Description	Tests that Java annotations are not used in improper locations. This testcase has an @Property annotation on a method parameter where the method is not a constructor.
Artifacts	JCA_10001_TestCase.java Test_JCA_10001.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl service1BadAnnotation.java Service1.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10001_TestCase~~

~~Test Assertion~~	~~JCA-TA-10001~~
~~Description~~	~~Tests that Java annotations are not used in improper locations. This test-case has an @Property annotation on a method parameter where the method is not a constructor.~~
~~Artifacts~~	~~JCA_10001_TestCase.java~~ ~~Test_JCA_10001.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~service1BadAnnotation.java~~ ~~Service1.java~~
~~Expected output~~	

JCA_10002_TestCase

Testcase ID	JCA_10002_TestCase
Test Assertion	JCA-TA-10002
Description	Tests that Java annotations are not used on static methods. This testcase has an @Property annotation on a static setter method.
Artifacts	JCA_10002_TestCase.java Test_JCA_10002.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl service1StaticAnnotation.java Service1.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10002_TestCase~~
~~Test Assertion~~	~~JCA-TA-10002~~
~~Description~~	~~Tests that Java annotations are not used on static methods. This testcase has an @Property annotation on a static setter method.~~
~~Artifacts~~	~~JCA_10002_TestCase.java~~ ~~Test_JCA_10002.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~service1StaticAnnotation.java~~ ~~Service1.java~~
~~Expected output~~	

JCA_10003_TestCase

Testcase ID	JCA_10003_TestCase
Test Assertion	JCA-TA-10003
Description	Tests that Java annotations are not used on static fields. This testcase has an @Property annotation on a static field.
Artifacts	JCA_10003_TestCase.java Test_JCA_10003.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl service1StaticAnnotation.java Service1.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10003_TestCase~~
~~Test Assertion~~	~~JCA-TA-10003~~
~~Description~~	~~Tests that Java annotations are not used on static fields. This testcase has an @Property annotation on a static field.~~
~~Artifacts~~	~~JCA_10003_TestCase.java~~ ~~Test_JCA_10003.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~service1StaticAnnotation.java~~ ~~Service1.java~~
~~Expected output~~	

339

340 ## JCA_10004_TestCase

341

Testcase ID	JCA_10004_TestCase
Test Assertion	JCA-TA-10031
Description	Tests that the @Callback annotation has to be specified with any parameters when used as the injection point of a callback reference.
Artifacts	JCA_10004_TestCase.java Test_JCA_10004.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1CalbackImpl.java

	service3BadCallbackImpl.java Service3Callback.java Service3WithCallback.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10004_TestCase~~
~~Test Assertion~~	~~JCA-TA-10031~~
~~Description~~	~~Tests that the @Callback annotation has to be specified with any parameters when used as the injection point of a callback reference.~~
~~Artifacts~~	~~JCA_10004_TestCase.java Test_JCA_10004.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1CalbackImpl.java service3BadCallbackImpl.java Service3Callback.java Service3WithCallback.java~~
~~Expected output~~	

JCA_10005_TestCase

Testcase ID	JCA_10005_TestCase
Test Assertion	JCA-TA-10005
Description	Tests that @Constructor works correctly with annotated parameters.
Artifacts	JCA_10005_TestCase.java Test_JCA_10005.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1ConstrImpl.java
Expected output	Positive test: "JCA_10005 request service1 operation1 invoked"
~~Testcase ID~~	~~JCA_10005_TestCase~~
~~Test Assertion~~	~~JCA-TA-10005~~
~~Description~~	~~Tests that @Constructor works correctly with annotated parameters.~~
~~Artifacts~~	~~JCA_10005_TestCase.java Test_JCA_10005.composite TestInvocation.wsdl TestClient_0002.composite~~

	~~Service1.wsdl~~ ~~Service1.java~~ ~~service1ConstrImpl.java~~
~~Expected output~~	~~Positive:~~

JCA_10006_TestCase

Testcase ID	JCA_10006_TestCase
Test Assertion	JCA-TA-10005
Description	Tests that the runtime raises an error when an @Constructor annotated constructor does not have all of it's parameters annotated.
Artifacts	JCA_10006_TestCase.java Test_JCA_10006.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1BadConstrImpl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10006_TestCase~~
~~Test Assertion~~	~~JCA-TA-10005~~
~~Description~~	~~Tests that the runtime raises an error when an @Constructor annotated constructor does not have all of it's parameters annotated.~~
~~Artifacts~~	~~JCA_10006_TestCase.java~~ ~~Test_JCA_10006.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1BadConstrImpl.java~~
~~Expected output~~	

JCA_10007_TestCase

Testcase ID	JCA_10007_TestCase
Test Assertion	JCA-TA-10006
Description	Tests that the runtime raises an error when an @Destroy method has a non-void return and parameters.
Artifacts	JCA_10007_TestCase.java

	Test_JCA_10007.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1BadDestroyImpl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10007_TestCase~~
~~Test Assertion~~	~~JCA-TA-10006~~
~~Description~~	~~Tests that the runtime raises an error when an @Destroy method has a non-void return and parameters.~~
~~Artifacts~~	~~JCA_10007_TestCase.java~~ ~~Test_JCA_10007.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1BadDestroyImpl.java~~
~~Expected output~~	

JCA_10008_TestCase

Testcase ID	JCA_10008_TestCase
Test Assertion	JCA-TA-10009
Description	Tests that the runtime raises an error when an @Init method has a non-void return and parameters.
Artifacts	JCA_10008_TestCase.java Test_JCA_10008.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1BadInitImpl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10008_TestCase~~
~~Test Assertion~~	~~JCA-TA-10009~~
~~Description~~	~~Tests that the runtime raises an error when an @Init method has a non-~~

	~~void return and parameters.~~
~~Artifacts~~	~~JCA_10008_TestCase.java~~ ~~Test_JCA_10008.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1BadInitImpl.java~~
~~Expected output~~	

354

355 JCA_10009_TestCase

356

Testcase ID	JCA_10009_TestCase
Test Assertion	JCA-TA-10011
Description	Tests that the runtime raises an error when an @Property annotated field is marked as final.
Artifacts	JCA_10009_TestCase.java Test_JCA_10009.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1BadPropImpl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10009_TestCase~~
~~Test Assertion~~	~~JCA-TA-10011~~
~~Description~~	~~Tests that the runtime raises an error when an @Property annotated field is marked as final.~~
~~Artifacts~~	~~JCA_10009_TestCase.java~~ ~~Test_JCA_10009.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1BadPropImpl.java~~
~~Expected output~~	

357

358 JCA_10010_TestCase

359

Testcase ID	JCA_10010_TestCase
Test Assertion	JCA-TA-10012

Description	Tests that @Property specifies a name when used with @Constructor.
Artifacts	JCA_10010_TestCase.java Test_JCA_10010.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1ConstrBadPropImpl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10010_TestCase~~
~~Test Assertion~~	~~JCA-TA-10012~~
~~Description~~	~~Tests that @Property specifies a name when used with @Constructor.~~
~~Artifacts~~	~~JCA_10010_TestCase.java~~ ~~Test_JCA_10010.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1ConstrBadPropImpl.java~~
~~Expected output~~	

JCA_10011_TestCase

Testcase ID	JCA_10011_TestCase
Test Assertion	JCA-TA-10013
Description	Tests that @Property is not required=true when used with @Constructor.
Artifacts	JCA_10011_TestCase.java Test_JCA_10011.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1ConstrBadPropImpl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10011_TestCase~~
~~Test Assertion~~	~~JCA-TA-10013~~
~~Description~~	~~Tests that @Property is not required=true when used with @Constructor.~~

~~Artifacts~~	~~JCA_10011_TestCase.java~~ ~~Test_JCA_10011.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1ConstrBadPropImpl.java~~
~~Expected output~~	

363

364 JCA_10012_TestCase

365

Testcase ID	JCA_10012_TestCase
Test Assertion	JCA-TA-10014
Description	Tests that multi valued properties are introspected correctly.
Artifacts	JCA_10012_TestCase.java Test_JCA_10012.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1MultiPropImpl.java
Expected output	Positive test: "JCA_10012 request service1 service2 service3 operation1 invoked"
~~Testcase ID~~	~~JCA_10012_TestCase~~
~~Test Assertion~~	~~JCA-TA-10014~~
~~Description~~	~~Tests that multi valued properties are introspected correctly.~~
~~Artifacts~~	~~JCA_10012_TestCase.java~~ ~~Test_JCA_10012.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1MultiPropImpl.java~~
~~Expected output~~	~~Positive:~~

366

367 JCA_10013_TestCase

368

Testcase ID	JCA_10013_TestCase
Test Assertion	JCA-TA-10016
Description	Tests that intent annoations can have qualifiers.

Artifacts	JCA_10013_TestCase.java Test_JCA_10013.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1IntentQualifier.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10013_TestCase~~
~~Test Assertion~~	~~JCA-TA-10016~~
~~Description~~	~~Tests that intent annoations can have qualifiers.~~
~~Artifacts~~	~~JCA_10013_TestCase.java~~ ~~Test_JCA_10013.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1IntentQualifier.java~~
~~Expected output~~	~~Positive:~~

JCA_10014_TestCase

Testcase ID	JCA_10014_TestCase
Test Assertion	JCA-TA-10018
Description	Tests that @Reference has a name when used with @Constructor.
Artifacts	JCA_10014_TestCase.java Test_JCA_10014.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1ConstrBadRefImpl.java service1Impl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10014_TestCase~~

~~Test Assertion~~	~~JCA-TA-10018~~
~~Description~~	~~Tests that @Reference has a name when used with @Constructor.~~
~~Artifacts~~	~~JCA_10014_TestCase.java~~ ~~Test_JCA_10014.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1ConstrBadRefImpl.java~~ ~~service1Impl.java~~
~~Expected output~~	~~Negative:~~

372

373 ## JCA_10015_TestCase

374

Testcase ID	JCA_10015_TestCase
Test Assertion	JCA-TA-10019
Description	Tests that @Reference has required=true when used with @Constructor.
Artifacts	JCA_10015_TestCase.java Test_JCA_10015.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1ConstrBadRefImpl.java service1Impl.java
Expected output	Negative test: "exception"
~~Testcase ID~~	~~JCA_10015_TestCase~~
~~Test Assertion~~	~~JCA-TA-10019~~
~~Description~~	~~Tests that @Reference has required=true when used with @Constructor.~~
~~Artifacts~~	~~JCA_10015_TestCase.java~~ ~~Test_JCA_10015.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1ConstrBadRefImpl.java~~ ~~service1Impl.java~~
~~Expected output~~	~~Negative:~~

375

JCA_10016_TestCase

Testcase ID	JCA_10016_TestCase
Test Assertion	JCA-TA-10022
Description	Tests that @Reference(required=false) is introspected as multiplicity 0..n.
Artifacts	JCA_10016_TestCase.java Test_JCA_10016.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl Service1.java service1OptMultiRef.java
Expected output	Positive test: "JCA_10016 request service1 operation1 invoked"

JCA_10029_TestCase

Testcase ID	JCA_10029_TestCase
Test Assertion	JCA-TA-10059
Description	Tests that a @Remotable annotation on a method that is not a setter is detected
Artifacts	JCA_10029_TestCase.java Test_JCA_10029.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceBadRemotableMethod.java service1BadRemotableMethodCallerImpl.java service1BadRemotableMethodImpl.java
Expected output	Negative test: "exception"

JCA_10030_TestCase

Testcase ID	JCA_10030_TestCase
Test Assertion	JCA-TA-10059
Description	Tests that a @Remotable annotation on a parameter that is not a constructor parameter is detected
Artifacts	JCA_10030_TestCase.java Test_JCA_10030.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceBadRemotable.java service1BadRemotableCallerImpl.java service1BadRemotableImpl.java
Expected output	Negative test: "exception"

JCA_10031_TestCase

Testcase ID	JCA_10031_TestCase
Test Assertion	JCA-TA-10060
Description	Tests that a field annotated with @Callback is typed with an interface specified in the service interface class
Artifacts	JCA_10031_TestCase.java Test_JCA_10031.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3WithCallback.java Service3Callback.java Service3BadCallback.java service1CallbackImpl.java service3ImplBad.java
Expected output	Negative test:

	"exception"

JCA_10032_TestCase

Testcase ID	JCA_10032_TestCase
Test Assertion	JCA-TA-10061
Description	Tests that a method annotated @OneWay has a void return type
Artifacts	JCA_10032_TestCase.java Test_JCA_10032.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceOneWayNoVoid.java service1OneWayNoVoidCallerImpl.java ServiceOneWayNoVoidImpl.java
Expected output	Negative test: "exception"

JCA_10033_TestCase

Testcase ID	JCA_10033_TestCase
Test Assertion	JCA-TA-10061
Description	Tests that a method annotated @OneWay throws no exceptions
Artifacts	JCA_10033_TestCase.java Test_JCA_10033.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceOneWayThrows.java service1OneWayThrowsCallerImpl.java ServiceOneWayThrowsImpl.java
Expected output	Negative test:

	"exception"

JCA_10034_TestCase

Testcase ID	JCA_10034_TestCase
Test Assertion	JCA-TA-10034
Description	Tests that a Java interface containing a @org.oasisopen.sca.annotation.OneWay annotation is executed in a non-blocking manner
Artifacts	JCA_10034_TestCase.java Test_JCA_10034.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceOneWay.java service1OneWayCallerImpl2.java serviceOneWayImpl.java
Expected output	Positive test: "JCA_10034 request service1 operation1 invoked service2 operation1 invoked OneWay method inonly previously called with testInvoke"
~~Testcase ID~~	~~JCA_10016_TestCase~~
~~Test Assertion~~	~~JCA-TA-10022~~
~~Description~~	~~Tests that @Reference(required=false) is introspected as multiplicity 0..n.~~
~~Artifacts~~	~~JCA_10016_TestCase.java~~ ~~Test_JCA_10016.composite~~ ~~TestInvocation.wsdl~~ ~~TestClient_0002.composite~~ ~~Service1.wsdl~~ ~~Service1.java~~ ~~service1OptMultiRef.java~~
~~Expected output~~	~~Positive:~~

JCA_10035_TestCase

Testcase ID	JCA_10035_TestCase
Test Assertion	JCA-TA-10063
Description	Tests that a Composite scoped implementation does not contain a field annotated with @Callback

Artifacts	JCA_10035_TestCase.java Test_JCA_10035.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3WithCallback.java Service3Callback.java service1CallbackImpl.java service3ImplComposite.java
Expected output	Negative test: "exception"

JCA_10044_TestCase

Testcase ID	JCA_10044_TestCase
Test Assertion	JCA-TA-10024
Description	Tests that where a component reference with multiplicity of 0..1 is not wired, that the reference injected into the implementation is null
Artifacts	JCA_10044_TestCase.java Test_JCA_10044.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1Impl6.java
Expected output	Positive test: "JCA_10044 request service1 operation1 invoked reference is null"

JCA_10045_TestCase

Testcase ID	JCA_10045_TestCase
Test Assertion	JCA-TA-10025
Description	Tests that where a component reference with multiplicity of 0..n is not wired, that the reference injected into the implementation is an empty array

	or collection
Artifacts	JCA_10045_TestCase.java Test_JCA_10045.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1Impl4.java
Expected output	Positive test: "JCA_10045 request service1 operation1 invoked reference array is empty"

411

412 ## JCA_10046_TestCase

413

Testcase ID	JCA_10046_TestCase
Test Assertion	JCA-TA-10045
Description	Tests that where the Java interface of a service is marked remotable, the interface is translatable into a WSDL portType
Artifacts	JCA_10046_TestCase.java Test_JCA_10046.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Unmappable.java unmappableServiceImpl.java
Expected output	Negative test: "exception"

414

415 ## JCA_10047_TestCase

416

Testcase ID	JCA_10047_TestCase
Test Assertion	JCA-TA-10046
Description	Tests that the @Scope annotation is only valid when applied to a component implementation class
Artifacts	JCA_10047_TestCase.java

	Test_JCA_10047.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.wsdl
Expected output	Negative test: "exception"

417

418 ## JCA_10048_TestCase

419

Testcase ID	JCA_10048_TestCase
Test Assertion	JCA-TA-10047
Description	Tests that where a class is annotated with a @Service annotation which declares a set of services, that the implementation class implements all of the methods of all the declared service interfaces
Artifacts	JCA_10048_TestCase.java Test_JCA_10048.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.wsdl
Expected output	Negative test: "exception"

420

421 ## JCA_10049_TestCase

422

Testcase ID	JCA_10049_TestCase
Test Assertion	JCA-TA-10050
Description	Tests that where a @Service annotation has a @names attribute, that the number of entries in the array of @names is the same as the number of entries in the array of the @value attribute
Artifacts	JCA_10049_TestCase.java Test_JCA_10049.composite TestInvocation.wsdl

	TestClient_0002.composite TestComposite1.composite Service1.wsdl
Expected output	Negative test: "exception"

423

424 JCA_10050_TestCase

425

Testcase ID	JCA_10050_TestCase
Test Assertion	JCA-TA-10055
Description	Tests that where an implementation class declares 2 or more services, that each service interface class has a distinct Java simple name
Artifacts	JCA_10050_TestCase.java Test_JCA_10050.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.wsdl
Expected output	Negative test: "exception"

426

427 JCA_10051_TestCase

428

Testcase ID	JCA_10051_TestCase
Test Assertion	JCA-TA-10057
Description	Tests that a @Service annotation has at least one element in its @value array
Artifacts	JCA_10051_TestCase.java Test_JCA_10051.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.wsdl
Expected output	Negative test:

	"exception"

JCA_10052_TestCase

Testcase ID	JCA_10052_TestCase
Test Assertion	JCA-TA-10058
Description	Tests that all the entries in the @names array of a @Service annotation are unique
Artifacts	JCA_10052_TestCase.java Test_JCA_10052.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.wsdl
Expected output	Negative test: "exception"

2.8 Section 11

JCA_11001_TestCase

Testcase ID	JCA_11001_TestCase
Test Assertion	JCA-TA-11001
Description	Tests that a Java interface which does not have @WebService annotation which is mapped to WSDL by an SCA runtime is treated as if it did contain a @WebService annotation
Artifacts	JCA_11001_TestCase.java Test_JCA_11001.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.java Service1ImplWSService.wsdl

	service1Impl.java service1Impl2.java
Expected output	Positive test: "JCA_11001 request service1 operation1 invoked service2 operation1 invoked"

437

438 JCA_11002_TestCase

439

Testcase ID	JCA_11002_TestCase
Test Assertion	JCA-TA-11002
Description	Tests that a Java interface containing a @org.oasisopen.sca.annotation.OneWay annotation is mapped to WSDL as if it contained a @javax.jws.OneWay annotation
Artifacts	JCA_11002_TestCase.java Test_JCA_11002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceOneWay.java service1OneWayCallerImpl.java serviceOneWayImpl.java
Expected output	Positive test: "JCA_11002 request service1 operation1 invoked service2 operation1 invoked OneWay method inonly previously called with testInvoke"

440 JCA_11003_TestCase

441

Testcase ID	JCA_11003_TestCase
Test Assertion	JCA-TA-11003
Description	Tests that if a Java interface mapped from a WSDL interface by an SCA runtime contains a @WebService annotation, the interface is treated as if it had a @Remotable annotation
Artifacts	JCA_11003_TestCase.java Test_JCA_11003.composite TestInvocation.wsdl TestClient_0002.composite

	Service1.java Service1WS.java service1ImplWS.java
Expected output	Positive test: "JCA_11003 request service1 operation1 invoked"

JCA_11004_TestCase

Testcase ID	JCA_11004_TestCase
Test Assertion	JCA-TA-11004
Description	Tests that if a Java interface used as a service interface contains JAXB 2.1 datatypes and annotations, the SCA runtime is able to map the interface to WSDL.
Artifacts	JCA_11004_TestCase.java Test_JCA_11004.composite TestInvocation.wsdl TestClient_0002.composite Service1.wsdl ServiceJAXB.java service1JAXBCallerImpl.java serviceJAXBImpl.java
Expected output	Positive test: "JCA_11004 request service1 operation1 invoked service2 operation1 invoked JAXB parameters received"

JCA_11005_TestCase

Testcase ID	JCA_11005_TestCase
Test Assertion	JCA-TA-11006
Description	Tests that a Java interface used to declare an SCA service interface in an <interface.java/> element does not contain the additional client-side asynchronous polling and callback methods defined by JAX-WS.
Artifacts	JCA_11005_TestCase.java Test_JCA_11005.composite TestInvocation.wsdl TestClient_0002.composite Service1.java

	Service1WithAsyncMethods.java service1Impl.java
Expected output	Negative test: "exception"

JCA_11006_TestCase

Testcase ID	JCA_11006_TestCase
Test Assertion	JCA-TA-11006
Description	Tests that where a Java interface used to declare an SCA reference interface contains the additional client side asynchronous polling and callback methods defined by JAX-WS, that the interface is treated as valid and is used for the reference.
Artifacts	JCA_11006_TestCase.java Test_JCA_11006.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1WithAsyncMethods.java service1AsyncRefImpl.java service1Impl.java
Expected output	Positive test: "JCA_11006 request service1 operation1 invoked service2 operation1 invoked"

JCA_11007_TestCase

Testcase ID	JCA_11007_TestCase
Test Assertion	JCA-TA-11008
Description	Tests that where a Java implementation class has an SCA reference which uses an interface which contains the additional client side asynchronous polling and callback methods defined by JAX-WS, that the introspected reference element in the component type for that implementation has an interface that does not contain the additional asynchronous polling and callback methods.
Artifacts	JCA_11007_TestCase.java Test_JCA_11007.composite TestInvocation.wsdl

	TestClient_0002.composite Service1.java Service1WithAsyncMethods.java service1AsyncRefImpl.java service1Impl.java
Expected output	Positive test: "JCA_11007 request service1 operation1 invoked service2 operation1 invoked"

JCA_11008_TestCase

Testcase ID	JCA_11008_TestCase
Test Assertion	JCA-TA-11009
Description	Tests that where a Java implementation class has an SCA reference which uses an interface which contains the additional client side asynchronous polling and callback methods defined by JAX-WS, that the implementation class is able to use the asynchronous polling and callback methods with the semantics described in the JAX-WS 2.1 specification.
Artifacts	JCA_11008_TestCase.java Test_JCA_11008.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1WithAsyncMethods.java service1AsyncRefImpl.java service1Impl.java
Expected output	Positive test: "JCA_11008 request service1 operation1 invoked service2 operation1 invoked service3 operation1 invoked"

JCA_11009_TestCase

Testcase ID	JCA_11009_TestCase
Test Assertion	JCA-TA-11013
Description	Tests that where a Java implementation class has an @WebService annotation with the @endpointInterface attribute has its (Java) service interface defined by the interface referenced by the @endpointInterface attrib-

	ute
Artifacts	JCA_11009_TestCase.java Test_JCA_11009.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service3Operations.java Service1Calls3OperationsImpl.java Service3OperationsWebServiceImpl.java
Expected output	Positive test: "JCA_11009 request service1 operation1 invoked"

455

456 ## JCA_11010_TestCase

457

Testcase ID	JCA_11010_TestCase
Test Assertion	JCA-TA-11014
Description	Tests that where a Java service interface contains a @WebParam annotation with its @header attribute set to "true", that the interface is treated as if it had the SOAP intent applied
Artifacts	JCA_11010_TestCase.java Test_JCA_11010.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1Impl.java Service1WithWebParam.java definitions.xml (in contribution JCA_General_POJO)
Expected output	Negative test: "exception"

458

459 ## JCA_11011_TestCase

460

Testcase ID	JCA_11011_TestCase

Test Assertion	JCA-TA-11015
Description	Tests that where a Java service interface contains a @WebResult annotation with its @header attribute set to "true", that the interface is treated as if it had the SOAP intent applied
Artifacts	JCA_11011_TestCase.java Test_JCA_11011.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1Impl.java Service1WithWebResult.java definitions.xml (in contribution JCA_General_POJO)
Expected output	Negative test: "exception"

461

462 ## JCA_11012_TestCase

463

Testcase ID	JCA_11012_TestCase
Test Assertion	JCA-TA-11020
Description	Tests that where a Java service interface contains a @SOAPBinding annotation with its @header attribute set to "true", that the interface is treated as if it had the SOAP intent applied
Artifacts	JCA_11012_TestCase.java Test_JCA_11012.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1Impl.java Service1WithWebSoapBinding.java definitions.xml (in contribution JCA_General_POJO)
Expected output	Negative test: "exception"

464

465 JCA_11013_TestCase

466

Testcase ID	JCA_11013_TestCase
Test Assertion	JCA-TA-11016
Description	Tests that where a Java service implementation class contains a @ServiceMode annotation that the service interface is treated as if it had the SOAP intent applied
Artifacts	JCA_11013_TestCase.java Test_JCA_11013.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1WithServiceModeImpl.java definitions.xml (in contribution JCA_General_POJO)
Expected output	Negative test: "exception"

467

468 JCA_11014_TestCase

469

Testcase ID	JCA_11014_TestCase
Test Assertion	JCA-TA-11017
Description	Tests that where a Java interface used to the define the interface of an SCA service does not contains an @WebServiceClient annotation
Artifacts	JCA_11014_TestCase.java Test_JCA_11014.composite TestInvocation.wsdl TestClient_0002.composite Service1WithWebServiceClient.java service1Impl.java
Expected output	Negative test: "exception"

JCA_11015_TestCase

Testcase ID	JCA_11015_TestCase
Test Assertion	JCA-TA-11018
Description	Tests that where a Java service implementation class contains a @WebServiceProvider annotation, the class is treated as if it is annotated with an SCA @Remotable annotation
Artifacts	JCA_11015_TestCase.java Test_JCA_11015.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1WithWebServiceProviderImpl.java
Expected output	Positive test: "JCA_11015 request service1 operation1 invoked"

JCA_11016_TestCase

Testcase ID	JCA_11016_TestCase
Test Assertion	JCA-TA-11019
Description	Tests that where a Java service implementation class contains a @WebServiceProvider annotation and the @wsdlLocation attribute of that annotation is declared, referencing a WSDL document, that the class is treated as if its service interface is defined by the referenced WSDL document
Artifacts	JCA_11016_TestCase.java Test_JCA_11016.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1WithWebServiceProviderImpl.java
Expected output	Negative test: "exception"

476

477 JCA_11017_TestCase

478

Testcase ID	JCA_11017_TestCase
Test Assertion	JCA-TA-9066 JCA-TA-9068 JCA-TA-11009 JCA-TA-11010
Description	Tests that where a client component Java implementation has a reference that uses the async callback client API to invoke a service which is an "asyncInvocation" asynchronous service, where the response to an operation is received some arbitrary time after the request is made, the client is able to make the invocation successfully and get back the result - covers both a regular response and a fault response.
Artifacts	JCA_11017_TestCase.java Test_JCA_11017.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service1AsyncServer.java service1AsyncRefImpl.java Service1AsyncServerImpl.java
Expected output	Positive test: "JCA_11017 request service1 operation1 invoked Future returned service2 operation1 invoked asynchrono org.oasisopen.sca.test.BusinessFault1: service2 operation1 invoked asynchronously"

479

3 Cross Mapping of Test Assertions to TestCases

Test Assertion	Test Cases
JCA-TA-2001	JCA_2001_TestCase
JCA-TA-2002	JCA_2002_TestCase
JCA-TA-2003	JCA_2003_TestCase
JCA-TA-2004	JCA_2004_TestCase
JCA-TA-2005	JCA_2005_TestCase
JCA-TA-2006	JCA_2006_TestCase
JCA-TA-2007	JCA_2007_TestCase
JCA-TA-2008	JCA_2008_TestCase
JCA-TA-2009	Optional - not tested
JCA-TA-2010	JCA_2009_TestCase
JCA-TA-2011	JCA_2010_TestCase

Test Assertion	Test Cases
JCA-TA-3001	JCA_3001_TestCase JCA_3002_TestCase
JCA-TA-3002	JCA_3003_TestCase JCA_3004_TestCase
JCA-TA-3003	JCA_3005_TestCase JCA_3006_TestCase JCA_3007_TestCase
JCA-TA-3004	JCA_3008_TestCase
JCA-TA-3005	JCA_3009_TestCase JCA_3010_TestCase
JCA-TA-3006	JCA_3011_TestCase
JCA-TA-3007	JCA_3012_TestCase
JCA-TA-3008	JCA_3013_TestCase
JCA-TA-3009	JCA_3014_TestCase

JCA_11018_TestCase

Test Assertion	Test Cases
JCA-TA-4001	JCA_4001_TestCase
JCA-TA-4002	JCA_4001_TestCase
JCA-TA-4003	JCA_4001_TestCase
JCA-TA-4004	JCA_4002_TestCase
JCA-TA-4005	JCA_4001_TestCase
JCA-TA-4006	JCA_4001_TestCase
JCA-TA-4007	JCA_4001_TestCase
JCA-TA-4008	JCA_4001_TestCase
JCA-TA-4009	JCA_4001_TestCase
JCA-TA-4010	JCA_4003_TestCase
JCA-TA-4011	JCA_4001_TestCase

~~JCA-TA-4012~~	~~JCA_4004_TestCase~~
~~JCA-TA-4013~~	~~JCA_4001_TestCase~~
~~JCA-TA-4014~~	~~JCA_4001_TestCase~~
~~JCA-TA-4015~~	~~JCA_4005_TestCase~~
~~JCA-TA-4016~~	~~JCA_4001_TestCase~~
~~JCA-TA-4017~~	~~JCA_4001_TestCase~~
~~JCA-TA-4018~~	~~JCA_4001_TestCase~~
~~JCA-TA-4019~~	~~untestable~~
~~JCA-TA-4020~~	~~JCA_4001_TestCase~~
~~JCA-TA-4021~~	~~JCA_4001_TestCase~~
~~JCA-TA-4022~~	~~JCA_4007_TestCase~~
~~JCA-TA-4023~~	~~JCA_4001_TestCase~~
~~JCA-TA-4024~~	~~JCA_4008_TestCase~~

485

Testcase ID	JCA_11018_TestCase
Test Assertion	JCA-TA-11022
Description	Tests that where a Java interface or class has an @WebService annotation with the @name attribute set, the name of an SCA interface defined by the Java interface for an implementation class with no @Service annotation is the value of the @name attribute
Artifacts	JCA_11018_TestCase.java Test_JCA_11018.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceWithName.java service1CallsWithNameImpl.java serviceWithNameImpl.java
Expected output	Positive test: "JCA_11018 request service1 operation1 invoked service2 operation1 invoked"

486

487 ## JCA_11019_TestCase

~~Test Assertion~~	~~Test Cases~~
~~JCA-TA-7001~~	~~JCA_7001_TestCase~~
~~JCA-TA-7002~~	~~JCA_7002_TestCase~~
~~JCA-TA-7003~~	~~JCA_7003_TestCase~~ ~~JCA_7004_TestCase~~
~~JCA-TA-7004~~	~~JCA_7003_TestCase~~
~~JCA-TA-7005~~	~~JCA_7003_TestCase~~
~~JCA-TA-7006~~	~~JCA_7005_TestCase~~

	JCA_7006_TestCase
JCA-TA-7007	JCA_7003_TestCase

488

Testcase ID	JCA_11019_TestCase
Test Assertion	JCA-TA-11023
Description	Tests that where a Java interface or class has an @WebMethod annotation with the @operationName attribute set the name of the corresponding operation in the SCA interface defined by the Java interface is the value * of the @operationName attribute
Artifacts	JCA_11019_TestCase.java Test_JCA_11019.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceWithOpName.java ServiceWithOpNameClient.java service1CallsWithOpNameImpl.java serviceWithOpNameImpl.java
Expected output	Positive test: "JCA_11019 request service1 operation1 invoked service2 operation1 invoked"

Test Assertion	Test Cases
JCA-TA-8001	JCA_8001_TestCase
JCA-TA-8002	JCA_8002_TestCase JCA_8005_TestCase
JCA-TA-8003	JCA_8003_TestCase JCA_8005_TestCase
JCA-TA-8004	JCA_8004_TestCase JCA_8005_TestCase
JCA-TA-8005	JCA_8006_TestCase
JCA-TA-8006	JCA_8007_TestCase
JCA-TA-8007	JCA_8008_TestCase
JCA-TA-8008	JCA_8009_TestCase
JCA-TA-8009	JCA_8010_TestCase
JCA-TA-8010	JCA_8011_TestCase

489

490 JCA_11020_TestCase

491

Testcase ID	JCA_11020_TestCase

Test Assertion	JCA-TA-11024
Description	Tests that where a Java method has an @WebMethod annotation with the @exclude attribute set to "true" the name of an method is not an operation in the SCA interface defined by the Java interface
Artifacts	JCA_11020_TestCase.java Test_JCA_11020.composite TestInvocation.wsdl TestClient_0002.composite Service1.java Service2Operations.java Service1Calls2OperationsImpl.java Service2OperationsImpl.java
Expected output	Negative test: "exception"

Test Assertion	Test Cases
JCA-TA-9001	JCA_9001_TestCase JCA_9008_TestCase
JCA-TA-9002	JCA_9002_TestCase
JCA-TA-9003	JCA_9003_TestCase
JCA-TA-9004	JCA_9004_TestCase
JCA-TA-9005	JCA_9005_TestCase
JCA-TA-9006	JCA_9006_TestCase
JCA-TA-9007	JCA_9006_TestCase
JCA-TA-9008	JCA_9006_TestCase
JCA-TA-9009	JCA_9007_TestCase
JCA-TA-9010	JCA_9008_TestCase
JCA-TA-9011	JCA_9008_TestCase
JCA-TA-9012	JCA_9008_TestCase
JCA-TA-9013	JCA_9008_TestCase
JCA-TA-9014	JCA_9006_TestCase
JCA-TA-9015	JCA_9009_TestCase
JCA-TA-9016	JCA_9009_TestCase
JCA-TA-9017	JCA_9009_TestCase
JCA-TA-9018	JCA_9009_TestCase
JCA-TA-9019	JCA_9009_TestCase
JCA-TA-9020	JCA_9010_TestCase
JCA-TA-9021	JCA_9010_TestCase
JCA-TA-9022	JCA_9010_TestCase
JCA-TA-9023	JCA_9010_TestCase
JCA-TA-9024	JCA_9010_TestCase
JCA-TA-9025	JCA_9011_TestCase
JCA-TA-9026	JCA_9011_TestCase
JCA-TA-9027	JCA_9012_TestCase

~~JCA-TA-9028~~	~~JCA_9012_TestCase~~
~~JCA-TA-9029~~	~~JCA_9012_TestCase~~
~~JCA-TA-9030~~	~~JCA_9013_TestCase~~
~~JCA-TA-9031~~	~~JCA_9013_TestCase~~
~~JCA-TA-9032~~	~~JCA_9013_TestCase~~
~~JCA-TA-9033~~	~~JCA_9013_TestCase~~
~~JCA-TA-9034~~	~~JCA_9014_TestCase~~
~~JCA-TA-9035~~	~~JCA_9014_TestCase~~
~~JCA-TA-9036~~	~~Untestable - no standard means of configuring a service to have a JAAS subject~~
~~JCA-TA-9037~~	~~Untestable - no standard means of configuring a service to have a JAAS subject~~
~~JCA-TA-9038~~	~~Untestable - no standard means of configuring a service to have a JAAS subject~~
~~JCA-TA-9039~~	~~JCA_9015_TestCase~~
~~JCA-TA-9040~~	~~JCA_9015_TestCase~~
~~JCA-TA-9041~~	~~JCA_9015_TestCase~~
~~JCA-TA-9042~~	~~JCA_9015_TestCase~~
~~JCA-TA-9043~~	~~JCA_9015_TestCase~~
~~JCA-TA-9044~~	~~JCA_9015_TestCase~~
~~JCA-TA-9045~~	~~JCA_9015_TestCase~~
~~JCA-TA-9046~~	~~JCA_9015_TestCase~~
~~JCA-TA-9047~~	~~JCA_9015_TestCase~~
~~JCA-TA-9048~~	~~JCA_9015_TestCase~~
~~JCA-TA-9049~~	~~JCA_9015_TestCase~~
~~JCA-TA-9050~~	~~JCA_9014_TestCase~~
~~JCA-TA-9051~~	~~JCA_9016_TestCase~~
~~JCA-TA-9052~~	~~JCA_9016_TestCase~~
~~JCA-TA-9053~~	~~JCA_9016_TestCase~~
~~JCA-TA-9054~~	~~JCA_9016_TestCase~~
~~JCA-TA-9055~~	~~JCA_9016_TestCase~~
~~JCA-TA-9056~~	~~JCA_9016_TestCase~~
~~JCA-TA-9057~~	~~JCA_9016_TestCase~~
~~JCA-TA-9058~~	~~JCA_9016_TestCase~~
~~JCA-TA-9059~~	~~JCA_9016_TestCase~~
~~JCA-TA-9060~~	~~JCA_9016_TestCase~~
~~JCA-TA-9061~~	~~Impossible to obtain a SCAClientFactory instance with an invalid domainURI - untestable~~
~~JCA-TA-9062~~	~~Untestable since this is a protected method.~~
~~JCA-TA-9063~~	~~Impossible to obtain a SCAClientFactory instance with an invalid domainURI - untestable~~
~~JCA-TA-9064~~	~~Untestable unless we make the vendor implementation of SCAClientFactoryFinder a conformance target~~
~~JCA-TA-9065~~	~~Untestable unless we make the vendor implementation of SCAClientFactoryFinder a conformance target~~
~~JCA-TA-9066~~	~~JCA_7003_TestCase~~

~~JCA-TA-9067~~	~~JCA_7005_TestCase~~
~~JCA-TA-9068~~	~~JCA_7006_TestCase~~
~~JCA-TA-9069~~	~~JCA_7006_TestCase~~
~~JCA-TA-9070~~	~~JCA_9006_TestCase~~

JCA_11021_TestCase

Testcase ID	JCA_11021_TestCase
Test Assertion	JCA-TA-11025
Description	Tests that where a Java parameter has an @WebParam annotation with the @mode attribute set that the parameter value is passed in the proper direction
Artifacts	JCA_11021_TestCase.java Test_JCA_11021.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceWithMode.java service1CallsWithModeImpl.java serviceWithModeImpl.java
Expected output	Positive test: "JCA_11021 request service1 operation1 invoked service2 operation1 invoked ok"

~~Test Assertion~~	~~Test Cases~~
~~JCA-TA-10001~~	~~JCA_10001_TestCase~~
~~JCA-TA-10002~~	~~JCA_10002_TestCase~~
~~JCA-TA-10003~~	~~JCA_10003_TestCase~~
~~JCA-TA-10004~~	~~JCA_2009_TestCase~~ ~~JCA_2010_TestCase~~
~~JCA-TA-10031~~	~~JCA_10004_TestCase~~
~~JCA-TA-10005~~	~~JCA_4001_TestCase~~ ~~JCA_10005_TestCase~~ ~~JCA_10006_TestCase~~
~~JCA-TA-10006~~	~~JCA_4001_TestCase~~ ~~JCA_10007_TestCase~~
~~JCA-TA-10007~~	~~JCA_4001_TestCase~~
~~JCA-TA-10008~~	~~JCA_2005_TestCase~~
~~JCA-TA-10009~~	~~JCA_10008_TestCase~~
~~JCA-TA-10010~~	~~JCA_4001_TestCase~~
~~JCA-TA-10011~~	~~JCA_10009_TestCase~~
~~JCA-TA-10012~~	~~JCA_10005_TestCase~~

	~~JCA_10010_TestCase~~
~~JCA-TA-10013~~	~~JCA_10005_TestCase~~ ~~JCA_10011_TestCase~~
~~JCA-TA-10014~~	~~JCA_10012_TestCase~~
~~JCA-TA-10015~~	~~JCA_2001_TestCase~~
~~JCA-TA-10016~~	~~JCA_10013_TestCase~~
~~JCA-TA-10017~~	~~JCA_2001_TestCase~~
~~JCA-TA-10018~~	~~JCA_10014_TestCase~~
~~JCA-TA-10019~~	~~JCA_10015_TestCase~~
~~JCA-TA-10020~~	~~JCA_2001_TestCase~~
~~JCA-TA-10021~~	~~JCA_2001_TestCase~~
~~JCA-TA-10022~~	~~JCA_10016_TestCase~~
~~JCA-TA-10023~~	~~JCA_2008_TestCase~~
~~JCA-TA-10024~~	~~JCA_10044_TestCase~~
~~JCA-TA-10025~~	~~JCA_10045_TestCase~~
~~JCA-TA-10026~~	~~optional requirement - no test~~
~~JCA-TA-10027~~	~~optional requirement - no test~~
~~JCA-TA-10028~~	~~optional requirement - no test~~
~~JCA-TA-10029~~	~~optional requirement - no test~~
~~JCA-TA-10030~~	~~requires undeploy API - no test~~
~~JCA-TA-10031~~	~~no TA~~
~~JCA-TA-10032~~	~~no TA~~
~~JCA-TA-10033~~	~~requires redeploy API - no test~~
~~JCA-TA-10034~~	~~optional requirement - no test~~
~~JCA-TA-10035~~	~~requires undeploy API - no test~~
~~JCA-TA-10036~~	~~requires undeploy API - no test~~
~~JCA-TA-10037~~	~~requires redeploy API - no test~~
~~JCA-TA-10038~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10039~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10040~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10041~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10042~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10043~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10044~~	~~requires optional support for rewiring - no test~~
~~JCA-TA-10045~~	~~JCA_10046_TestCase~~
~~JCA-TA-10046~~	~~JCA_10047_TestCase~~
~~JCA-TA-10047~~	~~JCA_10048_TestCase~~
~~JCA-TA-10050~~	~~JCA_10049_TestCase~~
~~JCA-TA-10055~~	~~JCA_10050_TestCase~~
~~JCA-TA-10056~~	~~requires deploy/redeploy API - no test~~
~~JCA-TA-10057~~	~~JCA_10051_TestCase~~
~~JCA-TA-10058~~	~~JCA_10052_TestCase~~

JCA_11022_TestCase

Testcase ID	JCA_11022_TestCase

Test Assertion	JCA-TA-11026
Description	Tests that where a Java interface or class used to define a service has an @WebService annotation with the @name attribute set and the implementation class has a @Service annotation with the name attribute specified, that the name for the service specified in the @Service annotation is used.
Artifacts	JCA_11022_TestCase.java Test_JCA_11022.composite TestInvocation.wsdl TestClient_0002.composite Service1.java ServiceWithName.java service1CallsWithNameImpl.java serviceWithNameImpl2.java
Expected output	Positive test: "JCA_11021 request service1 operation1 invoked service2 operation1 invoked"

~~Test Assertion~~	~~Test Cases~~
~~JCA-TA-11001~~	~~JCA_11001_TestCase~~
~~JCA-TA-11002~~	~~JCA_11002_TestCase~~
~~JCA-TA-11003~~	~~JCA_11003_TestCase~~
~~JCA-TA-11004~~	~~JCA_11004_TestCase~~
~~JCA-TA-11005~~	~~JCA_11005_TestCase~~
~~JCA-TA-11006~~	~~JCA_11006_TestCase~~
~~JCA-TA-11007~~	~~optional - no test case~~
~~JCA-TA-11008~~	~~JCA_11007_TestCase~~
~~JCA-TA-11009~~	~~JCA_11008_TestCase~~
~~JCA-TA-11010~~	~~JCA_7003_TestCase~~ ~~JCA_7004_TestCase~~
~~JCA-TA-11011~~	~~JCA_11003_TestCase~~
~~JCA-TA-11012~~	~~JCA_3014_TestCase~~
~~JCA-TA-11013~~	~~JCA_11009_TestCase~~
~~JCA-TA-11014~~	~~JCA_11010_TestCase~~
~~JCA-TA-11015~~	~~JCA_11011_TestCase~~
~~JCA-TA-11016~~	~~JCA_11013_TestCase~~
~~JCA-TA-11017~~	~~JCA_11014_TestCase~~
~~JCA-TA-11018~~	~~JCA_11015_TestCase~~
~~JCA-TA-11019~~	~~JCA_11016_TestCase~~
~~JCA-TA-11020~~	~~JCA_11012_TestCase~~
~~JCA-TA-11021~~	~~Tested by a range of the testeases in this test suite~~ ~~- default mappings~~ ~~- mappings influenced by JAXWS annotations~~ ~~- JAXB type mapping~~ ~~are all dealt with in various different testeases, so there is no need to create a new specific testease~~

	~~for this general requirement.~~

498

4 Catalog of Test Artifacts

4.1 Composite Files - lower level

Name	Valid	Description
Marked "Impl Specific" if it is implementation type specific		*Services and References use interface Service1 unless described otherwise*
CompositeScope.composite	**Y**	Composite containing a component with a composite scope implementation (service1CompositeImpl.java) invoked by a variety of other components
TestClient_002.composite	**Y**	Test invocation composite that presents the TestInvocation interface to the test runner client.

4.2 Java Interfaces

Name	Description
Constants.java	Policy related constants declared in an interface
DataStore.java	Data store interface used as service interface for data store component
MultipleService.java	Interface with a large number of operations, used to check multiple invocations to a service component
ParallelService.java	
Service1.java	Interface with 1 operation - "operation1", string input, string output
Service1AsyncServer.java	SCA Async Server form of the Service1 interface
Service1Superset.java	A superset of the Service1 interface - "operation1", string input, string output - "operation2", string input, string output
Service1WithAsyncMethods.java	Service1 service interface

	- this version has all the additional JAXWS async client methods added
Service1WithFaults.java	Version of the Service1 interface with a series of declared business faults on operation1
Service1WithFaultsAsyncServer.java	Service1 service interface with with a series of declared business faults on operation1 - Asynchronous server version
Service1WS.java	Service1 service interface - a variant which has a @WebService annotation - and no @Remotable annotation
Service2.java	A test service interface, designed to be incompatible with the Service1 interface 1 operation "operation2", int input, int output
Service2Operations.java	A service with 2 operations - operation1: input String response String, excluded via @WebMethod(exclude=ture) - operation2: input String response String
Service3BadCallback.java	A remotable service interface not used for a Callback from Service3 1 operation "callback1", string input, string output
Service3Callback.java	A remotable service interface used for a Callback from Service3 1 operation "callback1", string input, string output
Service3Operations.java	A service with 3 operations - operation1: input String response String - operation2: input String response String - operation3: input String response String
Service3OperationsWSDL.java	A service with 3 operations - with a @WebService annotation that references a WSDL - operation1: input String response String - operation2: input String response String - operation3: input String response String
Service3WithCallback.java	A bi-directional service interface with callback interface Service3Callback - 1 operation "operation3", string input, string output
Service4.java	A remotable service interface containing 1 opera-

	tion with a (Java) mutable parameter 1 operation "operation1" StringBuffer input, string output
Service5Intents.java	1 operation "operation1", string input, string output - entire interface is annotated with TestIntent4 - operation1 is annotated with TestIntent3
Service6PolicySets.java	Service interface - 1 operation "operation1", string input, string output - interface is annotated with policy set PolicySet1 - operation is annotated with policy set PolicySet2
Service7Callback.java	Remotable service interface used for a Callback from Service7 - 1 operation "service7Callback1" string input, string output
Service7WithCallback.java	A bi-directional service interface with callback interface Service7Callback 1 operation "operation7", string input, string output
ServiceBadRemotable.java	ServiceBadRemotable service interface 2 operations - “opeation2” has a parameter annotated @Remotable - "operation1" a Request/Response method, string input, string output
ServiceBadRemotableMethod.java	ServiceBadRemotableMethod service interface 2 operations - “opeation2” is annotated @Remotable - "operation1" a Request/Response method, string input, string output
ServiceJAXB.java	ServiceJAXB service interface - 1 operation "operation1", JAXB_Class1 input, JAXB_Class2 output This interface has JAXB classes as parameters
ServiceLocalRemote.java	Local service interface with a remotable callback interface (ServiceLocalRemoteCallback) 1 operation with 1 input, 1 output parameter
ServiceLocalRemoteCallback.java	Callback interface for ServiceLocalRemote - remot-

	able 1 operation with 1 input, 1 output parameter
ServiceOneWay.java	ServiceOneWay service interface 2 operations: - "inonly1" a OneWay method, string input - "operation1" a Request/Response method, string input, string output
ServiceOneWayMissing.java	ServiceOneWay service interface but with the in-only1 operation missing its @OneWay annotation 2 operations - "inonly1" a OneWay method missing its @One-Way annotation, string input - "operation1" a Request/Response method, string input, string output
ServiceOneWayNoVoid.java	ServiceOneWayNoVoid service interface 2 operations: - "operation2" is annotated @OneWay but has string output - "operation1" a Request/Response method, string input, string output
ServiceOneWayThrows.java	ServiceOneWayThrows service interface 2 operations: - "operation2" is annotated @OneWay but throws an exception - "operation1" a Request/Response method, string input, string output
ServiceRemoteLocal.java	Remotable service interface with a local callback interface (ServiceRemoteLocalCallback) 1 operation with 1 input, 1 output parameter
ServiceRemoteLocalCallback.java	Callback interface for ServiceRemoteLocal - local 1 operation with 1 input, 1 output parameter
ServiceWithName.java	ServiceWithName service interface - has a @WebService annotation with the name at-tribute set
ServiceWithMode.java	ServiceWithMode service interface

	- has a @WebParam annotation with the mode attribute set
ServiceWithOpName.java	ServiceWithOpName service interface - has a @WebMethod annotation with the operationName attribute set
ServiceWithOpNameClient.java	ServiceWithOpName service interface - directly uses the operation name specified in the @WebMethod annotation in ServiceWithOpName
TestInvocation.java	Basic interface to invoke testcases 1 operation - "invokeTest", string input, string output
TestIntent1.java	Intent annotation for test:testIntent1
TestIntent2.java	Intent annotation for test:testIntent2
TestIntent3.java	Intent annotation for test:testIntent3
TestIntent4.java	Intent annotation for test:testIntent4
TestIntent5.java	Intent annotation for test:testIntent5
TestIntent6.java	Intent annotation for test:testIntent6
~~Name~~	~~Description~~
~~Constants.java~~	~~Policy related constants declared in an interface~~
~~DataStore.java~~	~~Data store interface used as service interface for data store component~~
~~MultipleService.java~~	~~Interface with a large number of operations, used to check multiple invocations to a service component~~
~~ParallelService.java~~	
~~Service1.java~~	~~Interface with 1 operation~~ ~~- "operation1", string input, string output~~
~~Service1AsyncServer.java~~	~~SCA Async Server form of the Service1 interface~~
~~Service1Superset.java~~	~~A superset of the Service1 interface~~ ~~- "operation1", string input, string output~~ ~~- "operation2", string input, string output~~

Service1WithAsyncMethods.java	Service1 service interface - this version has all the additional JAXWS async client methods added
Service1WithFaults.java	Version of the Service1 interface with a series of declared business faults on operation1
Service1WithFaultsAsyncServer.java	Service1 service interface with with a series of declared business faults on operation1 - Asynchronous server version
Service1WS.java	Service1 service interface - a variant which has a @WebService annotation - and no @Remotable annotation
Service2.java	A test service interface, designed to be incompatible with the Service1 interface 1 operation "operation2", int input, int output
Service3Callback.java	A remotable service interface used for a Callback from Service3 1 operation "callback1", string input, string output
Service3Operations.java	A service with 3 operations - operation1: input String response String - operation2: input String response String - operation3: input String response String
Service3OperationsWSDL.java	A service with 3 operations - with a @WebService annotation that references a WSDL - operation1: input String response String - operation2: input String response String - operation3: input String response String
Service3WithCallback.java	A bi-directional service interface with callback interface Service3Callback - 1 operation "operation3", string input, string output
Service4.java	A remotable service interface containing 1 operation with a (Java) mutable parameter 1 operation "operation1" StringBuffer input, string output
Service5Intents.java	1 operation "operation1", string input, string output - entire interface is annotated with TestIntent4 - operation1 is annotated with TestIntent3

Service6PolicySets.java	Service interface - 1 operation "operation1", string input, string output - interface is annotated with policy set PolicySet1 - operation is annotated with policy set PolicySet2
Service7Callback.java	Remotable service interface used for a Callback from Service7 - 1 operation "service7Callback1" string input, string output
Service7WithCallback.java	A bi-directional service interface with callback interface Service7Callback 1 operation "operation7", string input, string output
ServiceJAXB.java	ServiceJAXB service interface - 1 operation "operation1", JAXB_Class1 input, JAXB_Class2 output This interface has JAXB classes as parameters
ServiceOneWay.java	ServiceOneWay service interface 2 operations: - "inonly1" a OneWay method, string input - "operation1" a Request/Response method, string input, string output
ServiceOneWayMissing.java	ServiceOneWay service interface but with the inonly1 operation missing its @OneWay annotation 2 operations - "inonly1" a OneWay method missing its @OneWay annotation, string input - "operation1" a Request/Response method, string input, string output
TestInvocation.java	Basic interface to invoke testcases 1 operation - "invokeTest", string input, string output
TestIntent1.java	Intent annotation for test:testIntent1
TestIntent2.java	Intent annotation for test:testIntent2
TestIntent3.java	Intent annotation for test:testIntent3
TestIntent4.java	Intent annotation for test:testIntent4

~~TestIntent5.java~~	~~Intent annotation for test:testIntent5~~
~~TestIntent6.java~~	~~Intent annotation for test:testIntent6~~

505 ## 4.3 Java Implementation Classes

506

Name	Description
	Services and references use interface Service1 unless otherwise described
ASM_0002_Client.java	1 service implementing TestInvocation 0 references
asyncControllerImpl.java	1 service 2 references (1..1) - reference1 with interface Service1 - dataStore with interface DataStore Acts as an invoker for an Async service and retrieves results from the DataStore
asyncControllerWithFaultsImpl.java	1 service 2 references (1..1) - reference1 with interface Service1WithFaults - dataStore with interface DataStore Acts as an invoker for an Async service which can throw Faults and retrieves results from the DataStore
BusinessFault1.java	A business fault class
BusinessFault2.java	A business fault class
compositeEagerInitImpl.java	1 service 1 reference (1..1) Implementation is COMPOSITE scope and is marked with @EagerInit - reference is invoked from method marked with @Init
dataStoreCompositeImpl.java	1 service with DataStore interface 0 references COMPOSITE scope - acts as a store for data under a key

lifecycleControllerImpl.java	1 service 2 references - reference1 (1..1) with interface Service1 - dataStore (1..1) with interface DataStore Acts as a controller component for a class that undergoes a series of lifecycle tests - the results are stored into a data store component and are retrieved at the end by this implementation
MultipleCallbacksImpl.java	3 services - Service1 with interface Service1 - Service3WithCallback with interface Service3WithCallback - Service7WithCallback with interface Service7WithCallback 0 references
multipleServiceClientImpl.java	1 service 1 reference - reference1 (1..1) with interface MultipleService Invokes all the operations of MultipleService in sequence for a single reference
multipleServiceImpl.java	1 service with interface MultipleService 0 references Implementation with many service operations that checks if a stateless instance ever gets invoked more than once
parallelCompositeClientImpl.java	1 service with interface Service1 1 reference 2 properties - "invocationCount" - number of parallel invocations - "maxWaitTime" - max time to wait for a response from the reference invocation - "reference1" (1..1) with interface ParallelService reference1 is invoked multiple times in parallel on multiple threads
parallelCompositeServiceImpl.java	1 service with interface ParallelService 0 references

	Implementation has COMPOSITE scope
parallelServiceClientImpl.java	1 service with interface Service1 1 reference 2 properties - "invocationCount" - number of parallel invocations - "maxWaitTime" - max time to wait for a response from the reference invocation - "reference1" (1..1) with interface ParallelService reference1 is invoked multiple times in parallel on multiple threads
parallelServiceImpl.java	1 service with interface ParallelService 0 references Implementation has STATELESS scope
service1AsyncRefImpl.java	1 service with interface Service1 1 reference (1..1) "reference1" with JAXWS async client form of Service1 inerface 2 properties - "invokeMethods" - array of strings indicating how to invoke methods on reference1 - "invokeParm" - array of strings which are the inpit parameters to the invocations in invokeMethods
Service1AsyncServerImpl.java	1 service with interface Service1AsyncServer 0 references Async service implementation of Service1, with response sent back on a separate thread.
Service1AsyncServerMultiFaultImpl.java	1 service with interface Service1WithFaultsAsyncServer 1 reference (1..1) with interface DataStore Async service implementation of Service1WithFaults, which calls ResponseDispatch.sendFault() method more than once
Service1AsyncServerMultiResponseImpl.java	1 service with interface Service1AsyncServer 1 reference (1..1) with interface DataStore Async service implementation of Service1, which calls ResponseDispatch.sendResponse() method more than once

service1BadRemotableCaller.java	1 service with interface Service1 1 reference with interface ServiceBadRemotable
service1BadRemotableMethodCaller.java	1 service with interface Service1 1 reference with interface ServiceBadMethodRe-motable
service1CallbackContextImpl1.java	1 service with interface Service1 1 reference (1..1) with interface Service3WithCall-back which has a Callback interface Service3Call-back Reference gets called when service1 operation1 is invoked
service1CallbackContextImpl2.java	1 service with interface Service1 1 reference (1..1) with interface Service3WithCall-back which has a Callback interface Service3Call-back The callback operation's method accesses the getRequestContext() method of the ComponentContext API during the processing of a callback method and checks that returned RequestContext is not null
service1CallbackImpl.java	1 service with interface Service1, 1 reference (1..1) with interface Service3WithCall-back which has a Callback interface Service3Call-back which is implemented by this class Reference gets called when service1 operation1 is invoked
Service1Calls2OperationsImpl.java	1 service with interface Service1, 1 reference (1..1) with interface Service2Opera-tions
Service1Calls3OperationsImpl.java	1 service with interface Service1, 1 reference (1..1) with interface Service3Opera-tions
Service1CallsWithModeImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceWithMode
Service1CallsWithNameImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceWithName
Service1CallsWithOpNameImpl.java	1 service with interface Service1,

	1 reference (1..1) with interface ServiceWithOp-NameClient
Service1CCcastImpl.java	1 service with interface Service1 1 reference "reference1" with interface Service1 Implementation performs invocations of the cast() method of the ComponentContext API
Service1CCcreateSelfReferenceAImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the create-SelfReference(Class<B>) method of the Compon-entContext API
Service1CCcreateSelfReferenceBImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the create-SelfReference(Class<B>, String) method of the ComponentContext API
Service1CCgetPropertyImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the get-Property() method of the ComponentContext API
Service1CCgetServiceImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the getSer-vice() method of the ComponentContext API
Service1CCgetServiceReferenceImpl.java	Java component implementation with 1 service with interface Service1 3 references - requiredRef (1..1) with interface Service1 - singledRef (0..1) with interface Service1 (intended to be unwired) - multiRef (0..n) with interface Service1 Implementation performs invocations of the getSer-viceReference() method of the ComponentContext API
Service1CCgetServiceReferencesImpl.java	Java component implementation with 1 service with interface Service1

	4 references - requiredRef (1..1) with interface Service1 - singledRef (0..1) with interface Service1 (intended to be unwired) - multiRef (0..n) with interface Service1 - multiRef2 (0..n) with interface Service1 (intended to be unwired) Implementation performs invocations of the getServiceReferences() method of the ComponentContext API
Service1CCgetServicesImpl.java	Java component implementation with 1 service with interface Service1 4 references - requiredRef (1..1) with interface Service1 - singledRef (0..1) with interface Service1 (intended to be unwired) - multiRef (0..n) with interface Service1 - multiRef2 (0..n) with interface Service1 (intended to be unwired) Implementation performs invocations of the getServices() method of the ComponentContext API
Service1CCgetURIImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the getURI() method of the ComponentContext API
service1CompositeImpl.java	1 service with interface Service1 0 references Implementation is COMPOSITE scope
service1ConstrImpl.java	1 service with interface Service1 0 references Implementation has constructor with parameter annotated with @Property
service1ContextImpl1.java	1 service with interface Service1 1 reference (1..n) with interface Service1 all configured wires get called when service1 operation1 is invoked

	Iimplementation invokes the ComponentContext getService() API
service1CoordinatorImpl.java	1 service with interface Service1 1 reference (1..n) with interface Service1 all configured wires get called when service1 operation1 is invoked Implementation acts as a coordinator between each of the invocations - it sends different data as input to each reference invocation - it records all the output data from all the invocations
service1GoodIntent.java	1 service with interface Service1 3 references Implementation has all references annotated with TestIntent2
service1Impl.java	1 service with interface Service1 0 references
service1Impl2.java	1 service with interface Service1, 1 reference (1..1) with interface Service1 reference gets called when service1 operation1 is invoked
service1Impl4.java	1 service with interface Service1 1 reference (0..n) with interface Service1 all configured wires get called when service1 operation1 is invoked - reports an empty reference array if it is unwired
service1Impl6.java	1 service with interface Service1 1 reference (0..1) with interface Service1 configured wire gets called when service1 operation1 is invoked, if present otherwise, this implementation reports its absence
service1Impl7.java	1 service with interface Service1, 1 reference (1..1) with interface Service4 - reference gets called when service1 operation1 is invoked - reference is marked with @AllowsPassByReference

service1Impl7b.java	1 service with interface Service1, 1 reference (1..1) with interface Service4 - reference gets called when service1 operation1 is invoked - reference is not marked @AllowsPassByReference and the implementation checks for alteration of input parameters
service1ImplWS.java	Java implementation - NOT used as an SCA component implementation, but instead used in the process of generating a WSDL interface through the standard JDK wsgen tool, which follows JAXWS 2.1 rules 1 service with interface Service1WS 0 references
service1InitCheckerImpl.java	1 service with interface Service1 1 reference (1..1) with interface Service1 2 properties - invocationCount - expectedResponse The implementation invokes the reference invocationCount times and checks for the response expectedResponse
service1InitImpl.java	1 service with interface Service1 0 references Implementation has a method marked with @Init - the response from the operation1 method of Service1 differs depending on whether the @Init method has been called before the operation1 method is called
service1Intent.java	1 service with interface Service1 0 references Implementation class is annotated with TestIntent2
service1IntentQualifier.java	1 service with interface Service1 0 references Implementation class is annotated with TestIntent6.qual1
service1JAXBCallerImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceJAXB reference gets called when service1 operation1 is

	invoked
service1LifecycleExceptionsImpl.java	1 service with interface Service1 1 reference (1..1) with interface Service1 Java STATELESS component implementation which tests the overall Lifecycle sequence of an SCA Java implementation class
service1LocalRemoteCallbackImpl.java	1 service 1 reference with ServiceLocalRemote/ServiceLocalRemoteCallback
service1MultiPropImpl.java	1 service with interface Service1 0 references 1 property which is multi-valued
service1OneWayCallerImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceOneWay - reference OneWay operation gets called first when service1 operation1 is invoked - then reference request/response operation gets called
Service1OneWayMissingCallerImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceOneWayMissing - reference OneWay operation gets called first when service1 operation1 is invoked - then reference request/response operation gets called
service1OneWayNoVoidCaller.java	1 service with interface Service1 1 reference with interface ServiceOneWayNoVoid
service1OneWayThrowsCaller.java	1 service with interface Service1 1 reference with interface ServiceOneWayThrows
service1OptMultiRef.java	1 service with interface Service1 1 reference (0..n) with interface Service1
Service1RCgetServiceNameImpl.java	1 service with interface Service1 2 references - singleRef (1..1) with interface Service1 - callback3Ref (1..1) with interface Service3WithCallback

	Implementation performs invocations of the RequestContext getServiceName() method of the API
service1RemoteLocalCallbackImpl.java	1 service 1 reference with ServiceRemoteLocal/ServiceRemoteLocalCallback interfaces
service1RequestContextImpl1.java	1 service with interface Service1 0 references Implementation invokes the ComponentContext getRequestContext() API and checks that the returned value is a valid RequestContext
service1RequestContextImpl2.java	1 service with interface Service1 0 references This implementation invokes the getServiceReference() method of the RequestContext API and checks that the returned ServiceReference represents the Service offered by this component
Service1SCAClientImpl.java	1 service with interface Service1 1 reference "reference1" with interface Service1 Implementation performs invocations of the SCAClient API
service1StatelessLifecycleImpl.java	1 service with interface Service1 1 reference (1..1) with interface Service1 Java STATELESS component implementation which tests the overall Lifecycle sequence of an SCA Java implementation class
service1SupersetImpl.java	1 service with interface Service1Superset 0 references
service1UninitImpl.java	1 service with interface Service1 0 references Implementation refuses to initialize - it throws an exception from its @Init method
service1WSImpl.java	1 service with interface Service1WS 0 references
service2Impl.java	1 service with interface Service2 0 references
Service2OperationsImpl.java	1 service with interface Service2Operations

	0 references
service3Impl1.java	1 service with interface Service3WithCallback 1 callback using interface Service3Callback 0 references
service3ImplBad.java	1 service with interface Service3WithCallback Field is annotated with @Callback but has a type that is not a callback interface for Service3WithCallback
service3ImplComposite.java	1 service with interface Service3WithCallback Implementation has composite scope and a field annotated with @Callback
Service3OperationsWebServiceImpl.java	1 service with interface Service3Operations, but declared through an @WebService annotation 0 references
Service3OperationsWSDLImpl.java	1 service with interface Service3OperationsWSDL 0 references
service4Impl1.java	1 service with interface Service4, where the implementation method modifies the input parameter 0 references - service is marked with @AllowsPassByReference
service4Impl.java	1 service with interface Service4, where the implementation method modifies the input parameter 0 references
service5Impl2.java	1 service with interface Service1, 1 reference (1..1) with interface Service5Intents reference gets called when service1 operation1 is invoked
service5Impl.java	1 service with interface Service5Intents 0 references
service6Impl2.java	1 service with interface Service1, 1 reference (1..1) with interface Service6PolicySets reference gets called when service1 operation1 is invoked

service6Impl.java	1 service with interface Service6PolicySets 0 references
ServiceBadRemotableImpl.java	1 service with interface ServiceBadRemotable-Method 0 references
ServiceBadRemotableMethodImpl.java	1 service with interface ServiceBadRemotable 0 references
serviceJAXBImpl.java	1 service with interface ServiceJAXB 0 references invocation of operation1 causes the implementation to read data from the supplied JAXB_Class1 parameter and return data in all the fields of the returned JAXB_Class2 parameter
serviceLocalRemoteImpl.java	1 service with interface ServiceLocalRemote
serviceOneWayImpl.java	1 service with interface ServiceOneWay 0 references The implementation is COMPOSITE scope and is stateful - it remembers whether its OneWay method inonly1 was called - it reports this when its request/Response method is called
serviceOneWayImpl2.java	1 service with interface ServiceOneWay 0 references The implementation is COMPOSITE scope and is stateful - it remembers whether its OneWay method inonly1 was called - the OneWay methods waits for the request/response method to start before it completes - it reports this when its request/Response method is called
serviceOneWayNoVoidImpl.java	1 service with interface ServiceOneWayNoVoid 0 references
serviceOneWayThrowsImpl.java	1 service with interface ServiceOneWayThrows 0 references
serviceRemoteLocalImpl.java	1 service with interface ServiceRemoteLocal

ServiceWithModeImpl.java	1 service with interface ServiceWithMode 0 references
ServiceWithNameImpl.java	1 service with interface ServiceWithName 0 references
ServiceWithNameImpl2.java	1 service with interface ServiceWithName 0 references The implementation class has a @Service annotation
ServiceWithOpNameImpl.java	1 service with interface ServiceWithOpName 0 references
TestException.java	Exception thrown by SCA Test services
JAXB.JAXB_Class1.java	JAXB annotated complex class that is used in a service interface
JAXB.JAXB_Class2.java	JAXB annotated complex class that is used in a service interface
Name	**Description**
	Services and references use interface Service1 unless otherwise described
ASM_0002_Client.java	1 service implementing TestInvocation 0 references
asyncControllerImpl.java	1 service 2 references (1..1) - reference1 with interface Service1 - dataStore with interface DataStore Acts as an invoker for an Async service and retrieves results from the DataStore
asyncControllerWithFaultsImpl.java	1 service 2 references (1..1) - reference1 with interface Service1WithFaults - dataStore with interface DataStore Acts as an invoker for an Async service which can throw Faults and retrieves results from the DataStore

BusinessFault1.java	A business fault class
BusinessFault2.java	A business fault class
compositeEagerInitImpl.java	1 service 1 reference (1..1) Implementation is COMPOSITE scope and is marked with @EagerInit - reference is invoked from method marked with @Init
dataStoreCompositeImpl.java	1 service with DataStore interface 0 references COMPOSITE scope - acts as a store for data under a key
lifecycleControllerImpl.java	1 service 2 references - reference1 (1..1) with interface Service1 - dataStore (1..1) with interface DataStore Acts as a controller component for a class that undergoes a series of lifecycle tests - the results are stored into a data store component and are retrieved at the end by this implementation
MultipleCallbacksImpl.java	3 services - Service1 with interface Service1 - Service3WithCallback with interface Service3WithCallback - Service7WithCallback with interface Service7WithCallback 0 references
multipleServiceClientImpl.java	1 service 1 reference - reference1 (1..1) with interface MultipleService Invokes all the operations of MultipleService in sequence for a single reference
multipleServiceImpl.java	1 service with interface MultipleService 0 references Implementation with many service operations that checks if a stateless instance ever gets invoked more than once

parallelCompositeClientImpl.java	1 service with interface Service1 1 reference 2 properties - "invocationCount" - number of parallel invocations - "maxWaitTime" - max time to wait for a response from the reference invocation - "reference1" (1..1) with interface ParallelService reference1 is invoked multiple times in parallel on multiple threads
parallelCompositeServiceImpl.java	1 service with interface ParallelService 0 references Implementation has COMPOSITE scope
parallelServiceClientImpl.java	1 service with interface Service1 1 reference 2 properties - "invocationCount" - number of parallel invocations - "maxWaitTime" - max time to wait for a response from the reference invocation - "reference1" (1..1) with interface ParallelService reference1 is invoked multiple times in parallel on multiple threads
parallelServiceImpl.java	1 service with interface ParallelService 0 references Implementation has STATELESS scope
service1AsyncRefImpl.java	1 service with interface Service1 1 reference (1..1) "reference1" with JAXWS async client form of Service1 inerface 2 properties - "invokeMethods" - array of strings indicating how to invoke methods on reference1 - "invokeParm" - array of strings which are the inpit parameters to the invocations in invokeMethods
Service1AsyncServerImpl.java	1 service with interface Service1AsyncServer 0 references Async service implementation of Service1, with response sent back on a separate thread.

Service1AsyncServerMultiFaultImpl.java	1 service with interface Service1WithFaultsAsyncServer 1 reference (1..1) with interface DataStore Async service implementation of Service1WithFaults, which calls ResponseDispatch.sendFault() method more than once
Service1AsyncServerMultiResponseImpl.java	1 service with interface Service1AsyncServer 1 reference (1..1) with interface DataStore Async service implementation of Service1, which calls ResponseDispatch.sendResponse() method more than once
service1CallbackContextImpl1.java	1 service with interface Service1 1 reference (1..1) with interface Service3WithCallback which has a Callback interface Service3Callback Reference gets called when service1 operation1 is invoked
service1CallbackContextImpl2.java	1 service with interface Service1 1 reference (1..1) with interface Service3WithCallback which has a Callback interface Service3Callback The callback operation's method accesses the getRequestContext() method of the ComponentContext API during the processing of a callback method and checks that returned RequestContext is not null
service1CallbackImpl.java	1 service with interface Service1, 1 reference (1..1) with interface Service3WithCallback which has a Callback interface Service3Callback which is implemented by this class Reference gets called when service1 operation1 is invoked
Service1Calls3OperationsImpl.java	1 service with interface Service1, 1 reference (1..1) with interface Service3Operations
Service1CCcastImpl.java	1 service with interface Service1 1 reference "reference1" with interface Service1 Implementation performs invocations of the cast() method of the ComponentContext API
Service1CCcreateSelfReferenceAImpl.java	1 service with interface Service1

	0 references Implementation performs invocations of the createSelfReference(Class<B>) method of the ComponentContext API
Service1CCcreateSelfReferenceBImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the createSelfReference(Class<B>, String) method of the ComponentContext API
Service1CCgetPropertyImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the getProperty() method of the ComponentContext API
Service1CCgetServiceImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the getService() method of the ComponentContext API
Service1CCgetServiceReferenceImpl.java	Java component implementation with 1 service with interface Service1 3 references - requiredRef (1..1) with interface Service1 - singledRef (0..1) with interface Service1 (intended to be unwired) - multiRef (0..n) with interface Service1 Implementation performs invocations of the getServiceReference() method of the ComponentContext API
Service1CCgetServiceReferencesImpl.java	Java component implementation with 1 service with interface Service1 4 references - requiredRef (1..1) with interface Service1 - singledRef (0..1) with interface Service1 (intended to be unwired) - multiRef (0..n) with interface Service1 - multiRef2 (0..n) with interface Service1 (intended to be unwired)

	Implementation performs invocations of the getServiceReferences() method of the ComponentContext API
Service1CCgetServicesImpl.java	Java component implementation with 1 service with interface Service1 4 references - requiredRef (1..1) with interface Service1 - singledRef (0..1) with interface Service1 (intended to be unwired) - multiRef (0..n) with interface Service1 - multiRef2 (0..n) with interface Service1 (intended to be unwired) Implementation performs invocations of the getServices() method of the ComponentContext API
Service1CCgetURIImpl.java	1 service with interface Service1 0 references Implementation performs invocations of the getURI() method of the ComponentContext API
service1CompositeImpl.java	1 service with interface Service1 0 references Implementation is COMPOSITE scope
service1ConstrImpl.java	1 service with interface Service1 0 references Implementation has constructor with parameter annotated with @Property
service1ContextImpl1.java	1 service with interface Service1 1 reference (1..n) with interface Service1 all configured wires get called when service1 operation1 is invoked Implementation invokes the ComponentContext getService() API
service1CoordinatorImpl.java	1 service with interface Service1 1 reference (1..n) with interface Service1 all configured wires get called when service1 operation1 is invoked Implementation acts as a coordinator between each of the invocations

	- it sends different data as input to each reference invocation - it records all the output data from all the invocations
service1GoodIntent.java	1 service with interface Service1 3 references Implementation has all references annotated with TestIntent2
service1Impl.java	1 service with interface Service1 0 references
service1Impl2.java	1 service with interface Service1, 1 reference (1..1) with interface Service1 reference gets called when service1 operation1 is invoked
service1Impl4.java	1 service with interface Service1 1 reference (0..n) with interface Service1 all configured wires get called when service1 operation1 is invoked - reports an empty reference array if it is unwired
service1Impl6.java	1 service with interface Service1 1 reference (0..1) with interface Service1 configured wire gets called when service1 operation1 is invoked, if present otherwise, this implementation reports its absence
service1Impl7.java	1 service with interface Service1, 1 reference (1..1) with interface Service4 - reference gets called when service1 operation1 is invoked - reference is marked with @AllowsPassByReference
service1Impl7b.java	1 service with interface Service1, 1 reference (1..1) with interface Service4 - reference gets called when service1 operation1 is invoked - reference is not marked @AllowsPassByReference and the implementation checks for alteration of input parameters
service1ImplWS.java	Java implementation - NOT used as an SCA component implementation, but instead used in the

	process of generating a WSDL interface through the standard JDK wsgen tool, which follows JAXWS 2.1 rules 1 service with interface Service1WS 0 references
service1InitCheckerImpl.java	1 service with interface Service1 1 reference (1..1) with interface Service1 2 properties - invocationCount - expectedResponse The implementation invokes the reference invocationCount times and checks for the response expectedResponse
service1InitImpl.java	1 service with interface Service1 0 references Implementation has a method marked with @Init - the response from the operation1 method of Service1 differs depending on whether the @Init method has been called before the operation1 method is called
service1Intent.java	1 service with interface Service1 0 references Implementation class is annotated with TestIntent2
service1IntentQualifier.java	1 service with interface Service1 0 references Implementation class is annotated with TestIntent6.qual1
service1JAXBCallerImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceJAXB reference gets called when service1 operation1 is invoked
service1LifecycleExceptionsImpl.java	1 service with interface Service1 1 reference (1..1) with interface Service1 Java STATELESS component implementation which tests the overall Lifecycle sequence of an SCA Java implementation class
service1MultiPropImpl.java	1 service with interface Service1

	0 references 1 property which is multi-valued
service1OneWayCallerImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceOneWay - reference OneWay operation gets called first when service1 operation1 is invoked - then reference request/response operation gets called
Service1OneWayMissingCallerImpl.java	1 service with interface Service1, 1 reference (1..1) with interface ServiceOneWay-Missing - reference OneWay operation gets called first when service1 operation1 is invoked - then reference request/response operation gets called
service1OptMultiRef.java	1 service with interface Service1 1 reference (0..n) with interface Service1
Service1RCgetServiceNameImpl.java	1 service with interface Service1 2 references - singleRef (1..1) with interface Service1 - callback3Ref (1..1) with interface Service3WithC-allback Implementation performs invocations of the Re-questContext getServiceName() method of the API
service1RequestContextImpl1.java	1 service with interface Service1 0 references Implementation invokes the ComponentContext getRequestContext() API and checks that the re-turned value is a valid RequestContext
service1RequestContextImpl2.java	1 service with interface Service1 0 references This implementation invokes the getServiceRefer-ence() method of the RequestContext API and checks that the returned ServiceReference repres-ents the Service offered by this component
Service1SCAClientImpl.java	1 service with interface Service1 1 reference "reference1" with interface Service1

	~~Implementation performs invocations of the SCAClient API~~
~~service1StatelessLifecycleImpl.java~~	~~1 service with interface Service1~~ ~~1 reference (1..1) with interface Service1~~ ~~Java STATELESS component implementation which tests the overall Lifecycle sequence of an SCA Java implementation class~~
~~service1SupersetImpl.java~~	~~1 service with interface Service1Superset~~ ~~0 references~~
~~service1UninitImpl.java~~	~~1 service with interface Service1~~ ~~0 references~~ ~~Implementation refuses to initialize - it throws an exception from its @Init method~~
~~service1WSImpl.java~~	~~1 service with interface Service1WS~~ ~~0 references~~
~~service2Impl.java~~	~~1 service with interface Service2~~ ~~0 references~~
~~service3Impl1.java~~	~~1 service with interface Service3WithCallback~~ ~~1 callback using interface Service3Callback~~ ~~0 references~~
~~Service3OperationsWebServiceImpl.java~~	~~1 service with interface Service3Operations, but declared through an @WebService annotation~~ ~~0 references~~
~~Service3OperationsWSDLImpl.java~~	~~1 service with interface Service3OperationsWSDL~~ ~~0 references~~
~~service4Impl1.java~~	~~1 service with interface Service4, where the implementation method modifies the input parameter~~ ~~0 references~~ ~~- service is marked with @AllowsPassByReference~~
~~service4Impl.java~~	~~1 service with interface Service4, where the implementation method modifies the input parameter~~ ~~0 references~~
~~service5Impl2.java~~	~~1 service with interface Service1,~~

	~~1 reference (1..1) with interface Service5Intents~~ ~~reference gets called when service1 operation1 is invoked~~
~~service5Impl.java~~	~~1 service with interface Service5Intents~~ ~~0 references~~
~~service6Impl2.java~~	~~1 service with interface Service1,~~ ~~1 reference (1..1) with interface Service6PolicySets~~ ~~reference gets called when service1 operation1 is invoked~~
~~service6Impl.java~~	~~1 service with interface Service6PolicySets~~ ~~0 references~~
~~serviceJAXBImpl.java~~	~~1 service with interface ServiceJAXB~~ ~~0 references~~ ~~invocation of operation1 causes the implementation to read data from the supplied JAXB_Class1 parameter and return data in all the fields of the returned JAXB_Class2 parameter~~
~~serviceOneWayImpl.java~~	~~1 service with interface ServiceOneWay~~ ~~0 references~~ ~~The implementation is COMPOSITE scope and is stateful~~ ~~- it remembers whether its OneWay method inonly1 was called~~ ~~- it reports this when its request/Response method is called~~
~~TestException.java~~	~~Exception thrown by SCA Test services~~
~~JAXB.JAXB_Class1.java~~	~~JAXB annotated complex class that is used in a service interface~~
~~JAXB.JAXB_Class2.java~~	~~JAXB annotated complex class that is used in a service interface~~

4.4 WSDL Interface Files

Name	Description
Service1ImplWSService.wsdl	WSDL mapped from service1ImplWS class
Service3OperationsWSDL.wsdl	WSDL version of the Service3OperationsWSDL.-java interface, but containing only operations "oper-

	ation1" and "operation2" - "operation3" intentionally missing
TestClient.wsdl	WSDL version of the TestInvocation.java interface
TestInvocation.wsdl	WSDL version of the client TestInvocation.java interface

509

5 Conformance

The artifacts contained in the sca-j-caa-1.1-testcases-cd01.zip file are considered to be authoritative and take precedence over the artifacts described in this document~~re are no conformance statements relating to the TestCases~~.

An implementation that claims to conform to this specification MUST be able to run all test cases in Section 2 TestCases, producing the 'Expected Output'.

Appendix A. Test Assertions for SCA Java Common Annotations and APIs Specification Version 1.1.

This document defines the Test Assertions for the SCA Java Common Annotations and APIs Specification Version 1.1.

The test assertions in this document follow the format defined in the OASIS Test Assertion Guidelines specification [TA-GUIDE].

Appendix A.1 Example Test Assertion

Test assertions are presented in a tabular format with rows corresponding to the entry types defined in [TA-GUIDE].

Assertion ID	JCA-TA-xxxx
Source	[JCAx00yy]
Target	<kitchenSink/> element of composite file
Prerequisites	The <kitchenSink/> element has a @drain attribute
Predicate	The @drain attribute value of the <kitchenSink/> element is a URI that identifies a portal into the drainage system of the Domain.
Prescription Level	Mandatory
Tags	kitchenSink drain Domain

Assertion ID: Is a unique ID for the test assertion. Its format starts with a 3 letter string that identifies the specification to which it relates - "JCA" is for the SCA Java Common Annotations and APIs specification. This is followed by "-TA-" to indicate that this identifier is for a test assertion. This is then followed by a unique 4 digit number.

Source: Is the identifier(s) of the normative statement(s) in the specification to which this assertion relates.

Target: Identifies the target which is addressed by this assertion. This is typically some SCA document element, or other SCA artifact but possibly could identify an SCA runtime and its behaviour.

Prerequisites: Defines any prerquisites for this test assertion. The prerequisites can be defined in terms of one or more other test assertions that have to be true.

Predicate: The meat of the assertion - something that evaluates to true or false for the given target.

Prescription Level: Mandatory (for MUST requirements) or Preferred (for SHOULD requirements) or Permitted (for MAY requirements).

Tags: Zero or more labels that are attached to this test assertion - these tags can be used to group sets of assertions.

Appendix A.2 Test Assertions for SCA Java CAA Specification Section 2

Assertion ID	JCA-TA-2001

Source	[JCA20001]
Target	Java service marked remotable
Prerequisites	
Predicate	Service does not use method overloading
Prescription Level	mandatory
Tags	"remotable" "overloading"

545

546

Assertion ID	JCA-TA-2002
Source	[JCA20002]
Target	Stateless scoped Java implementation instance
Prerequisites	
Predicate	Implementation instance is dispatched on a single thread at one time
Prescription Level	mandatory
Tags	"stateless scope" "single thread"

547

548

Assertion ID	JCA-TA-2003
Source	[JCA20003]
Target	Stateless scoped Java implementation instance
Prerequisites	
Predicate	Implementation instance is invoked once through one business method during the lifetime of the implementation instance
Prescription Level	mandatory
Tags	"stateless scope" "lifetime" "invocation"

549

550

Assertion ID	JCA-TA-2004
Source	[JCA20004]
Target	Domain level component implemented by a Java class marked "composite scope"
Prerequisites	

Predicate	All clients of the component appear to interact with a single runtime instance of the implementation class
Prescription Level	mandatory
Tags	"composite scope" "invocation"

551

552

Assertion ID	JCA-TA-2005
Source	[JCA20005]
Target	Java implementation class marked with "composite scope" and with "eager initialization"
Prerequisites	Containing component is started
Predicate	The Java implementation class instance is created and initialized
Prescription Level	mandatory
Tags	"composite scope" "lifetime" "eager initialization"

553

554

Assertion ID	JCA-TA-2006
Source	[JCA20006]
Target	Method of Java implementation class with marked with @Init annotation
Prerequisites	Implementation instance is created
Predicate	Method is called
Prescription Level	mandatory
Tags	"composite scope" "lifetime" "initialization" "@Init"

555

556

Assertion ID	JCA-TA-2007
Source	[JCA20007]
Target	Instance of a Java implementation class marked "composite scope"
Prerequisites	Implementation class is used as a component implementation Multiple invocations of services offered by the component are made, which overlap in time
Predicate	Multiple threads run in the single instance of the implementation class
Prescription	optional

Level	
Tags	"composite scope" "lifetime" "eager initialization"

Assertion ID	JCA-TA-2008
Source	[JCA20008]
Target	Java implementation class marked with "composite scope"
Prerequisites	Implementation class is used to implement a component nested inside a composite used to implement a domain level component Component offers at least one service Component has one or more clients for its service(s)
Predicate	Clients of the component appear to interact with a single instance of the implementation class
Prescription Level	mandatory
Tags	"composite scope" "nested component" "instance"

Assertion ID	JCA-TA-2009
Source	[JCA20009]
Target	Client invocation of a method of a service implementation
Prerequisites	Service method implementation marked "allows pass by reference" Client reference proxy marked "allows pass by reference" Client and service implementation run in the same JVM
Predicate	The invocation uses pass-by-reference call semantics for the input and response parameters
Prescription Level	optional
Tags	"allows pass by reference" "pass by reference" "local"

Assertion ID	JCA-TA-2010
Source	[JCA20010]
Target	Client invocation of a service implementation
Prerequisites	Service implementation method not marked "allows pass by reference" Client and service implementation run in the same JVM

Predicate	The invocation uses pass-by-value semantics for the input and response parameters
Prescription Level	mandatory
Tags	"allows pass by reference" "pass by value "local"

562

563

Assertion ID	JCA-TA-2011
Source	[JCA20010]
Target	Client invocation of a service implementation
Prerequisites	Client reference is not marked "allows pass by reference" Client and service implementation run in the same JVM
Predicate	The invocation uses pass-by-value semantics for the input and response parameters
Prescription Level	mandatory
Tags	"allows pass by reference" "pass by value "local"

564

565 # Appendix A.3 Test Assertions for SCA Java CAA Specification
566 Section 3

567

Assertion ID	JCA-TA-3001
Source	[JCA30001]
Target	@interface attribute of a <interface.java/> element
Prerequisites	
Predicate	Value of @interface attribute is the fully qualified name of a Java interface class
Prescription Level	mandatory
Tags	"interface element" "interface attribute" "Java interface class"

568

569

Assertion ID	JCA-TA-3002
Source	[JCA30002]
Target	@callbackInterface attribute of <interface.java/> element
Prerequisites	interface.java element has a @callbackInterface attribute

Predicate	Value of @callbackInterface is the fully qualified name of a Java interface class
Prescription Level	mandatory
Tags	"interface element" "callbackInterface attribute" "Java interface class"

570

571

Assertion ID	JCA-TA-3003
Source	[JCA30003]
Target	@callbackInterface attribute of a <interface.java/> element
Prerequisites	interface.java element has a @callbackInterface attribute Java interface class referenced by the @interface attribute contains a @C-allback annotation which references a Java interface class
Predicate	Value of @callbackInterface attribute is the same Java interface class as the one referenced by the @Callback annotation of the Java interface class referenced by the @interface attribute
Prescription Level	mandatory
Tags	"interface element" "callbackInterface attribute" "Java interface class" "@Callback"

572

573

Assertion ID	JCA-TA-3004
Source	[JCA30004]
Target	<interface.java/> element
Prerequisites	
Predicate	Conforms to the sca-interface-java.xsd schema
Prescription Level	mandatory
Tags	"interface element" "schema"

574

575

Assertion ID	JCA-TA-3005
Source	[JCA30005]
Target	@remotable attribute on <interface.java/> element
Prerequisites	Java interface class referenced by @interface attribute of interface.java element contains a @Remotable annotation

Predicate	@remotable attribute has the value "true"
Prescription Level	mandatory
Tags	"interface element" "remotable attribute" "Java interface class" "@Remotable"

576

577

Assertion ID	JCA-TA-3006
Source	[JCA30006]
Target	Java interface class referenced by @interface attribute of a <interface.java/> element
Prerequisites	
Predicate	Java interface class does not contain any of the annotations: @AllowsPassByReference, @ComponentName, @Constructor, @Context, @Destroy, @EagerInit, @Init, @Intent, @Property, @Qualifier, @Reference, @Scope, @Service.
Prescription Level	mandatory
Tags	"interface element" "SCA annotations" "Java interface class"

578

579

Assertion ID	JCA-TA-3007
Source	[JCA30007]
Target	Java interface class referenced by @callbackInterface attribute of a <interface.java/> element
Prerequisites	
Predicate	Java interface class does not contain any of the annotations: @AllowsPassByReference, @Callback, @ComponentName, @Constructor, @Context, @Destroy, @EagerInit, @Init, @Intent, @Property, @Qualifier, @Reference, @Scope, @Service.
Prescription Level	mandatory
Tags	"interface element" "SCA annotations" "Java interface class"

580

581

Assertion ID	JCA-TA-3008
Source	[JCA30009]
Target	2 java interface classes, A and B

Prerequisites	a) Interface classes A is compatible with interface class B, or is a compatible superset of B or is a compatible subset of B b) One or more of the methods contained in A or in B is annotated with @OneWay
Predicate	Every method which is present in both interface A and in interface B and which is annotated with @OneWay in either interface class is also annotated with @OneWay in the other interface class.
Prescription Level	mandatory
Tags	"interface classes" "@OneWay" "compatible"

582

Assertion ID	JCA-TA-3009
Source	[JCA30010]
Target	<interface.java/> element within a composite
Prerequisites	a) element references a Java class which contains a @WebService annotation or a @WebServiceProvider annotation b) the annotation contains a non-empty wsdlLocation property pointing at a WSDL document
Predicate	<interface.java/> element is treated as if it were an <interface.wsdl/> element with an @interface attribute identifying the portType in the WSDL document mapped from the Java interface class and containing @requires and @policySets attributes equal to the @requires and @policySets attributes of the <interface.java/> element
Prescription Level	mandatory
Tags	"interface.java element" "@WebService" "@WebServiceProvider" "interface.wsdl element"

583

584

585 # Appendix A.4 Test Assertions for SCA Java CAA Specification
586 Section 4

587

Assertion ID	JCA-TA-4001
Source	[JCA40001]
Target	Constructor of Java component implementation class
Prerequisites	Component is at the start of the Constructing state of its lifecycle
Predicate	Constructor is invoked by the SCA runtime
Prescription Level	mandatory

Tags	"lifecycle" "constructing" "constructor"

588

589

Assertion ID	JCA-TA-4002
Source	[JCA40002]
Target	Constructor of Java component implementation class
Prerequisites	Constructor is invoked by the SCA runtime at the start of the Constructing state of its lifecycle
Predicate	Constructor references and properties are injected when the constructor is invoked
Prescription Level	mandatory
Tags	"lifecycle" "constructing" "constructor" "injection" "reference" "property"

590

591

Assertion ID	JCA-TA-4003
Source	[JCA40003]
Target	Java component implementation class in Constructing state
Prerequisites	Constructor method completes successfully
Predicate	Component implementation transitions to the Injecting state
Prescription Level	mandatory
Tags	"lifecycle" "constructing" "injecting" "transition"

592

Assertion ID	JCA-TA-4004
Source	[JCA40004]
Target	Java component implementation class in Constructing state
Prerequisites	Constructor method throws an exception
Predicate	Component implementation class is transitioned to the Terminating state
Prescription Level	mandatory
Tags	"lifecycle" "constructor" "exception" "terminating"

593

Assertion ID	JCA-TA-4005
Source	[JCA40005]

Target	Java component implementation class in Injecting state
Prerequisites	Implementation has one or more properties
Predicate	Properties are injected (via field or setter injection) before any other actions occur
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "property"

594

Assertion ID	JCA-TA-4006
Source	[JCA40006]
Target	Java component implementation class in Injecting state
Prerequisites	Implementation has one or more references Implementation has one or more properties
Predicate	References are injected (via field or setter injection) after properties have been injected
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "reference"

595

596

Assertion ID	JCA-TA-4007
Source	[JCA40007]
Target	Java component implementation with one or more injected property / reference
Prerequisites	Implementation code does not use synchronization
Predicate	Injected properties and references are visible to the component implementation
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "synchronization" "property" "reference"

597

Assertion ID	JCA-TA-4008
Source	[JCA40008]
Target	Java component implementation in injecting state
Prerequisites	Implementation has 1 or more service methods
Predicate	Service methods are not invoked

Prescription Level	mandatory
Tags	"lifecycle" "injecting" "service methods"

598

Assertion ID	JCA-TA-4009
Source	[JCA40009]
Target	Java component implementation in injecting state
Prerequisites	All properties and references have been injected
Predicate	Component implementation transitions to the Initializing state
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "transition" "initializing"

599

Assertion ID	JCA-TA-4010
Source	[JCA40010]
Target	Java component implementation in injecting state
Prerequisites	Exception is thrown while a property or a reference is being injected
Predicate	Component implementation transitions to the Destroying state
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "exception" "transition" "destroying"

600

Assertion ID	JCA-TA-4011
Source	[JCA40011]
Target	Java component implementation entering initializing state
Prerequisites	Implementation has a method annotated with @Init
Predicate	Method annotated with @Init is invoked
Prescription Level	mandatory
Tags	"lifecycle" "initializing" "@Init"

601

602

Assertion ID	JCA-TA-4012
Source	[JCA40012]
Target	Java component implementation in initializing state

Prerequisites	Implementation has an injected reference Target of injected reference has not been initialized Implementation invokes a method on the reference
Predicate	Implementation receives a ServiceUnavailableException
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "exception" "reference" "uninitialized"

603

Assertion ID	JCA-TA-4013
Source	[JCA40013]
Target	Java component implementation in initializing state
Prerequisites	Implementation has 1 or more service methods
Predicate	Service methods are not invoked
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "service" "invocation"

604

Assertion ID	JCA-TA-4014
Source	[JCA40014]
Target	Java component implementation in initializing state
Prerequisites	Invocation of method annotated with @Init completes successfully
Predicate	Component implementation transitions to the Running state
Prescription Level	mandatory
Tags	"lifecycle" "initializing" "transition" "running"

605

Assertion ID	JCA-TA-4015
Source	[JCA40015]
Target	Java component implementation in initializing state
Prerequisites	Exception is thrown by the component implementation
Predicate	Component implementation transitions to the Destroying state
Prescription Level	mandatory
Tags	"lifecycle" "initializing" "exception" "transition" "destroying"

606

Assertion ID	JCA-TA-4016
Source	[JCA40016]
Target	Java component implementation in running state
Prerequisites	Component has one or more service Client component invokes one of the service operations
Predicate	Implementation service method corresponding to the operation is invoked
Prescription Level	mandatory
Tags	"lifecycle" "running" "service" "invocation" "method"

607

608

Assertion ID	JCA-TA-4017
Source	[JCA40017]
Target	Java component implementation in running state
Prerequisites	Implementation scope ends
Predicate	Component implementation transitions to the Destroying state
Prescription Level	mandatory
Tags	"lifecycle" "running" "scope ends" "transition" "destroying"

609

Assertion ID	JCA-TA-4018
Source	[JCA40018]
Target	Java component implementation entering Destroying state
Prerequisites	Implementation has a method annotated with @Destroy
Predicate	Method annotated with @Destroy is invoked
Prescription Level	mandatory
Tags	"lifecycle" "destroying" "@Destroy" "transition"

610

Assertion ID	JCA-TA-4019
Source	[JCA40019]
Target	Java component implementation in destroying state
Prerequisites	Implementation has an injected reference Implementation invokes a method on the reference

	Target of reference has already been destroyed
Predicate	Implementation receives an InvalidServiceException
Prescription Level	mandatory
Tags	"lifecycle" "destroying" "exception" "reference"

611

Assertion ID	JCA-TA-4020
Source	[JCA40020]
Target	Java component implementation in destroying state
Prerequisites	Implementation has one or more services Client component invokes service operations
Predicate	Implementation service methods are not invoked
Prescription Level	mandatory
Tags	"lifecycle" "destroying" "service" "method" "invocation"

612

613

Assertion ID	JCA-TA-4021
Source	[JCA40021]
Target	Java component implementation in Destroying state
Prerequisites	Method with @Destroy annotation completes
Predicate	Component implementation transitions to terminated state
Prescription Level	mandatory
Tags	"lifecycle" "destroying" "@Destroy" "transition" "terminated"

614

Assertion ID	JCA-TA-4022
Source	[JCA40022]
Target	Java component implementation in Destroying state
Prerequisites	Method with @Destroy annotation throws an exception
Predicate	Component implementation transitions to the terminated state
Prescription Level	mandatory
Tags	"lifecycle" "destroying" "exception" "transition" "terminated" "@Destroy"

615

Assertion ID	JCA-TA-4023
Source	[JCA40023]
Target	Java component implementation is in Terminated state
Prerequisites	Component implementation has service Client component invokes operations on a service
Predicate	Implementation service methods are not invoked
Prescription Level	mandatory
Tags	"lifecycle" "terminated" "service" "invocation"

Assertion ID	JCA-TA-4024
Source	[JCA40024]
Target	Java component implementation in Injecting state
Prerequisites	A property or a reference is unable to be injected
Predicate	Component implementation transitions to the Destroying state
Prescription Level	mandatory
Tags	"lifecycle" "injecting" "property" "reference" "destroying"

Appendix A.5 Test Assertions for SCA Java CAA Specification Section 7

Assertion ID	JCA-TA-7001
Source	[JCA60001]
Target	fields and/or setter methods annotated with @Callback of a component implementation class which has a bidirectional service
Prerequisites	The bidirectional service is invoked
Predicate	A Callback reference object for the invoked service is injected into all fields and setter methods which have the type of the interface for the callback and null is injected into all fields and setter methods which have a different type
Prescription Level	mandatory
Tags	"@Callback" "field" "setter method" "injection" "callback reference"

Assertion ID	JCA-TA-7002

Source	[JCA60002]
Target	fields and/or setter methods annotated with @Callback of a component implementation class which has a bidirectional service
Prerequisites	Component implementation also has one or more non-bidirectional services One of the non-bidirectional services is invoked
Predicate	null is injected into all of the fields and setter methods annotated with @Callback
Prescription Level	mandatory
Tags	"@Callback" "field" "setter method" "injection"

622

Assertion ID	JCA-TA-7003
Source	[JCA60003]
Target	Java interface class which is an asynchronous mapping of WSDL portType with request/response operations
Prerequisites	
Predicate	Class is annotated with the "asyncInvocation" intent
Prescription Level	mandatory
Tags	

623

Assertion ID	JCA-TA-7004
Source	[JCA60003]
Target	Java interface class which is an SCA asynchronous service mapping of WSDL portType with request/response operations
Prerequisites	WSDL portType contains an operation with a name
Predicate	Class contains a method with a name that is the JAX-WS mapping of the portType operation name, with the added suffix "Async", which has: - a void return type - a set of parameters which match the JAX-WS mapping of the input parameters of the WSDL operation - an additional final parameter which is a ResponseDispatch object typed by the JAX-WS Response Bean mapping of the output parameter(s) of the WSDL operation
Prescription Level	mandatory
Tags	"SCA asynchronous service" "WSDL mapping"

Assertion ID	JCA-TA-7005
Source	[JCA60004]
Target	Interface of an SCA service is declared using an SCA asynchronous service interface
Prerequisites	
Predicate	The service is supported by the SCA runtime and is invocable.
Prescription Level	mandatory
Tags	"SCA asynchronous service"

Assertion ID	JCA-TA-7006
Source	[JCA60005]
Target	ResponseDispatch object passed to SCA service implementation when operation of an SCA asynchronous service interface is invoked
Prerequisites	Either the sendReponse method or the sendFault method of the ResponseDispatch object has already been called once
Predicate	An IllegalStateException is thrown when either the sendResponse method or the sendFault method is invoked
Prescription Level	mandatory
Tags	"ResponseDispatch" "sendResponse" "sendFault"

Assertion ID	JCA-TA-7007
Source	[JCA60006]
Target	Java interface of a service or of a reference
Prerequisites	Interface contains one or more methods characterized as "Asynchronous service methods"
Predicate	Java interface is treated for matching purposes as if the asynchronous service methods are mapped to the equivalent synchronous methods
Prescription Level	mandatory
Tags	"asynchronous service interface" "interface matching" "synchronous methods"

Appendix A.6 Test Assertions for SCA Java CAA Specification Section 8

Assertion ID	JCA-TA-8001
Source	[JCA70001]
Target	Declaration of a Java annotation that corresponds to an SCA intent
Prerequisites	
Predicate	Declaration includes the @Intent annotation
Prescription Level	mandatory
Tags	"annotation" "declaration" "@Intent" "intent"

631

632

Assertion ID	JCA-TA-8002
Source	[JCA70002]
Target	Method of Java implementation class
Prerequisites	Method is annotated with an Intent annotation
Predicate	Method is a setter method for an SCA reference, either through explicit marking with @Reference or through introspection as defined in component implementation specification
Prescription Level	mandatory
Tags	"annotation" "method" "intent"

633

634

Assertion ID	JCA-TA-8003
Source	[JCA70002]
Target	Field of a Java implementation class
Prerequisites	Field is annotated with an Intent annotation
Predicate	Field is an SCA reference field, either through explicit marking with @Reference or through introspection as defined in the component implementation specification
Prescription Level	mandatory
Tags	"annotation" "field" "intent"

635

Assertion ID	JCA-TA-8004
Source	[JCA70002]
Target	Constructor parameter of a Java implementation class

Prerequisites	Constructor parameter is annotated with an Intent annotation
Predicate	Constructor parameter is marked as an SCA reference using the @Reference annotation
Prescription Level	mandatory
Tags	"annotation" "constructor" "parameter" "intent"

636

Assertion ID	JCA-TA-8005
Source	[JCA70003]
Target	Java element annotated with multiple intent annotations
Prerequisites	
Predicate	The set of intents which apply to the Java element are the combination of all the separate intents declared by the annotations, following the combination rules defined in the SCA Policy Framework spec
Prescription Level	mandatory
Tags	"annotation" "multiple" "intent" "combination"

637

638

Assertion ID	JCA-TA-8006
Source	[JCA70004]
Target	Method of a Java interface
Prerequisites	One set of Intent annotations are specified on the Java interface A second set of Intent annotations are specified on the method
Predicate	The set of intents which apply to the method are the combination of the set of intents on the interface with the set of intents on the method, following the merging rules for a structural hierarchy defined in the SCA Policy Framework specification
Prescription Level	mandatory
Tags	"annotation" "multiple" "intent" "combination" "interface" "method"

639

640

Assertion ID	JCA-TA-8007
Source	[JCA70005]
Target	Method of Java implementation class
Prerequisites	Method is annotated with a PolicySets annotation

Predicate	Method is a setter method for an SCA reference, either through explicit marking with @Reference or through introspection as defined in component implementation specification
Prescription Level	mandatory
Tags	"annotation" "reference" "@PolicySets" "method"

641

642

Assertion ID	JCA-TA-8008
Source	[JCA70005]
Target	Field of a Java implementation class
Prerequisites	Field is annotated with a PolicySets annotation
Predicate	Field is an SCA reference field, either through explicit marking with @Reference or through introspection as defined in the component implementation specification
Prescription Level	mandatory
Tags	"annotation" "reference" "@PolicySets" "field"

643

644

Assertion ID	JCA-TA-8009
Source	[JCA70005]
Target	Constructor parameter of a Java implementation class
Prerequisites	Constructor parameter is annotated with a PolicySets annotation
Predicate	Constructor parameter is marked as an SCA reference using the @Reference annotation
Prescription Level	mandatory
Tags	"annotation" "constructor" "@PolicySets" "reference"

645

646

Assertion ID	JCA-TA-8010
Source	[JCA70006]
Target	Method of a Java interface
Prerequisites	Java interface is annotated with @PolicySets annotation Method is annotated with a second @PolicySets annotation
Predicate	The set of intents which apply to the method is the union of the policy sets

	specificed by the two @PolicySets annotations.
Prescription Level	mandatory
Tags	"annotation" "@PolicySets" "interface" "method" "combination"

Appendix A.7 Test Assertions for SCA Java CAA Specification Section 9

Assertion ID	JCA-TA-9001
Source	[JCA80001]
Target	invocation of getService(...) method of ComponentContext API
Prerequisites	referenceName parameter of the method identifies a reference of multiplicity 0..n or multiplicity 1..n
Predicate	getService(...) method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getService" "0..n" "1..n" "IllegalArgumentException"

Assertion ID	JCA-TA-9002
Source	[JCA80002]
Target	invocation of getRequestContext(...) method of ComponentContext API
Prerequisites	invocation takes place during the execution of a Java business method of a service operation invocation take place on the same Java thread used by the SCA runtime to invoke the business method
Predicate	invocation returns a non-null value for the RequestContext
Prescription Level	mandatory
Tags	"ComponentContext" "getRequestContext" "business method" "service"

Assertion ID	JCA-TA-9003
Source	[JCA80003]
Target	invocation of getServiceReference() method of RequestContext API
Prerequisites	invocation takes place during the execution of a Java business method of a

	service operation
Predicate	getServiceReference() method returns a ServiceReference object that represents the service that was invoked
Prescription Level	mandatory
Tags	"RequestContext" "service operation" "getServiceReference"

655

Assertion ID	JCA-TA-9004
Source	[JCA80003]
Target	invocation of getServiceReference() method of RequestContext API
Prerequisites	invocation takes place during the execution of a Java business method of a callback operation
Predicate	getServiceReference() method returns a ServiceReference object that represents the callback that was invoked
Prescription Level	mandatory
Tags	"RequestContext" "callback operation" "getServiceReference"

656

657

Assertion ID	JCA-TA-9005
Source	[JCA80002]
Target	invocation of getRequestContext(...) method of ComponentContext API
Prerequisites	invocation takes place during the execution of a Java business method of a callback operation invocation take place on the same Java thread use by the SCA runtime to invoke the business method
Predicate	invocation returns a non-null value for the RequestContext
Prescription Level	mandatory
Tags	"ComponentContext" "getRequestContext" "business method" "callback"

658

659

Assertion ID	JCA-TA-9006
Source	[JCA80005]
Target	invocation of ComponentContext.getServiceReference() method
Prerequisites	reference named by referenceName parameter does not have an interface of the type defined by the businessInterface parameter

Predicate	getServiceReference() method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReference()" "IllegalArgumentException" "interface type" "businessInterface"

660

Assertion ID	JCA-TA-9007
Source	[JCA80006]
Target	invocation of ComponentContext.getServiceReference() method
Prerequisites	component which is invoking the method does not have a reference with the name provided in the referenceName parameter
Predicate	getServiceReference() method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReference()" "IllegalArgumentException" "referenceName"

661

Assertion ID	JCA-TA-9008
Source	[JCA80007]
Target	invocation of ComponentContext.getServiceReference() method
Prerequisites	a) multiplicity of the reference named by the referenceName parameter is 0..1 b) reference has no target service configured
Predicate	getServiceReference() method returns null
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReference()" "IllegalArgumentException" "multiplicity" "target service"

662

Assertion ID	JCA-TA-9009
Source	[JCA80008]
Target	invocation of ComponentContext.getURI method
Prerequisites	
Predicate	returns the absolute URI of the component in the SCA Domain
Prescription Level	mandatory
Tags	"ComponentContext" "getURI"

663

Assertion ID	JCA-TA-9010
Source	[JCA80009]
Target	invocation of ComponentContext.getService method
Prerequisites	reference named by the referenceName parameter has a target service configured
Predicate	method returns a proxy object with the interface passed in the businessInterface parameter for the reference named in the referenceName parameter
Prescription Level	mandatory
Tags	"ComponentContext" "getService" "proxy"

664

Assertion ID	JCA-TA-9011
Source	[JCA80010]
Target	invocation of ComponentContext.getService method
Prerequisites	multiplicity of the reference named by the referenceName parameter is 0..1 reference has no target service configured
Predicate	method returns null
Prescription Level	mandatory
Tags	"ComponentContext" "getService" "0..1" "no target"

665

Assertion ID	JCA-TA-9012
Source	[JCA80011]
Target	invocation of ComponentContext.getService method
Prerequisites	the component does not have a reference with the name in the referenceName parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getService" "referenceName" "IllegalArgumentException"

666

Assertion ID	JCA-TA-9013
Source	[JCA80012]
Target	invocation of ComponentContext.getService method

Prerequisites	reference named by the referenceName parameter does not have an interface compatible with the interface supplied in the businessInterface parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getService" "businessInterface" "IllegalArgumentException"

667

Assertion ID	JCA-TA-9014
Source	[JCA80013]
Target	invocation of ComponentContext.getServiceReference method
Prerequisites	reference named by the referenceName parameter has a target service configured
Predicate	method returns a ServiceReference object with the interface passed in the businessInterface parameter for the reference named in the referenceName parameter
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReference" "ServiceReference"

668

Assertion ID	JCA-TA-9015
Source	[JCA80014]
Target	invocation of ComponentContext.getServices method
Prerequisites	reference identified by the referenceName parameter exists and is configured with one or more target services identified reference has the interface supplied in the businessInterface parameter
Predicate	returns a Collection containing one proxy object for each target service of the reference, each implementing the interface supplied in the businessInterface parameter
Prescription Level	mandatory
Tags	"ComponentContext" "getServices" "collection" "proxy"

669

Assertion ID	JCA-TA-9016
Source	[JCA80015]
Target	invocation of ComponentContext.getServices method
Prerequisites	reference identified by the referenceName parameter exists but is con-

	figured with zero target services
Predicate	returns an empty Collection
Prescription Level	mandatory
Tags	"ComponentContext" "getServices" "collection" "empty"

670

Assertion ID	JCA-TA-9017
Source	[JCA80016]
Target	invocation of ComponentContext.getServices method
Prerequisites	reference identified by the referenceName parameter has a multiplicity of either 0..1 or 1..1
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServices" "multiplicity" "0..1" "1..1" "IllegalArgumentException"

671

Assertion ID	JCA-TA-9018
Source	[JCA80017]
Target	invocation of ComponentContext.getServices method
Prerequisites	component does not have a reference with the name identified by the referenceName parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServices" "reference" "referenceName" "IllegalArgumentException"

672

Assertion ID	JCA-TA-9019
Source	[JCA80018]
Target	invocation of ComponentContext.getServices method
Prerequisites	reference identified by the referenceName parameter does not have an interface compatible with the interface supplied in the businessInterface parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServices" "reference" "businessInterface" "Illeg-

	alArgumentException"

673

Assertion ID	JCA-TA-9020
Source	[JCA80019]
Target	invocation of ComponentContext.getServiceReferences method
Prerequisites	reference identified by the referenceName parameter exists and is configured with one or more target services identified reference has the interface supplied in the businessInterface parameter
Predicate	returns a Collection containing one ServiceReference object for each target service of the reference, each implementing the interface supplied in the businessInterface parameter
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReferences" "collection" "ServiceReference"

674

Assertion ID	JCA-TA-9021
Source	[JCA80020]
Target	invocation of ComponentContext.getServiceReferencess method
Prerequisites	reference identified by the referenceName parameter exists and is configured with zero target services
Predicate	returns an empty Collection
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReferences" "collection" "ServiceReference" "unwired"

675

Assertion ID	JCA-TA-9022
Source	[JCA80021]
Target	invocation of ComponentContext.getServiceReferencess method
Prerequisites	reference identified by the referenceName parameter has multiplicity of 0..1 or 1..1
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReferences" "0..1" "1..1" "IllegalArgumentException"

676

Assertion ID	JCA-TA-9023
Source	[JCA80022]
Target	invocation of ComponentContext.getServiceReferencess method
Prerequisites	component does not have a reference with the name contained in the referenceName parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	ComponentContext" "getServiceReferences" "referenceName" "IllegalArgumentException"

677

Assertion ID	JCA-TA-9024
Source	[JCA80023]
Target	invocation of ComponentContext.getServiceReferencess method
Prerequisites	reference identified by the referenceName parameter does not have an interface compatible with the interface supplied in the businessInterface parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReferences" "businessInterface" "IllegalArgumentException"

678

Assertion ID	JCA-TA-9025
Source	[JCA80024]
Target	invocation of ComponentContext.createSelfReference(businessInterface) method
Prerequisites	invoking component has one or more services with the interface specified in the businessInterface parameter
Predicate	returns a ServiceReference object typed by the interface defined by the businessInterface parameter for one of the services of the invoking component which has that interface
Prescription Level	mandatory
Tags	"ComponentContext" "createSelfReference" "ServiceReference" "businessInterface"

679

Assertion ID	JCA-TA-9026

Source	[JCA80025]
Target	invocation of ComponentContext.createSelfReference(businessInterface) method
Prerequisites	invoking component has no service with the interface specified in the businessInterface parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "createSelfReference" "IllegalArgumentException" "businessInterface"

680

Assertion ID	JCA-TA-9027
Source	[JCA80026]
Target	invocation of ComponentContext.createSelfReference(businessInterface, serviceName) method
Prerequisites	invoking component has a service with the name specified in the serviceName parameter service with the name specified in serviceName has an interface
Predicate	returns a ServiceReference object typed by the interface defined by the businessInterface parameter for one of the services of the invoking component which has that interface
Prescription Level	mandatory
Tags	"ComponentContext" "createSelfReference" "serviceName" "ServiceReference" "businessInterface"

681

Assertion ID	JCA-TA-9028
Source	[JCA80027]
Target	invocation of ComponentContext.createSelfReference(businessInterface, serviceName) method
Prerequisites	invoking component has no service with the name specified in the serviceName parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "createSelfReference" "serviceName" "IllegalArgumentException"

682

Assertion ID	JCA-TA-9029

Source	[JCA80028]
Target	invocation of ComponentContext.createSelfReference(businessInterface, serviceName) method
Prerequisites	the service with the name specified in the serviceName parameter does not have the interface defined in the businessInterface parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "createSelfReference" "serviceName" "IllegalArgumentException" "businessInterface"

683

Assertion ID	JCA-TA-9030
Source	[JCA80029]
Target	invocation of ComponentContext.getProperty method
Prerequisites	component configuration contains a value for the property with name supplied in the propertyName parameter
Predicate	method returns an object of the type identified by the type parameter containing the value in the component configuration for the property with the name in the propertyName parameter
Prescription Level	mandatory
Tags	"ComponentContext" "getProperty" "property value" "propertyName"

684

Assertion ID	JCA-TA-9031
Source	[JCA80029]
Target	invocation of ComponentContext.getProperty method
Prerequisites	component configuration contains no value for the property with name supplied in the propertyName parameter
Predicate	method returns null
Prescription Level	mandatory
Tags	"ComponentContext" "getProperty" "property value" "propertyName" "null"

685

Assertion ID	JCA-TA-9032
Source	[JCA80030]
Target	invocation of ComponentContext.getProperty method
Prerequisites	component does not have a property with name supplied in the propertyName parameter

Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getProperty" "propertyName" "IllegalArgumentException"

686

Assertion ID	JCA-TA-9033
Source	[JCA80031]
Target	invocation of ComponentContext.getProperty method
Prerequisites	component property with the name supplied in the propertyName parameter does not have a type compatible with the supplied type parameter
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getProperty" "type" "IllegalArgumentException"

687

Assertion ID	JCA-TA-9034
Source	[JCA80032]
Target	invocation of ComponentContext.cast method
Prerequisites	
Predicate	method returns a ServiceReference object typed by the same interface as specified by the reference proxy object supplied in the target parameter
Prescription Level	mandatory
Tags	"ComponentContext" "cast" "ServiceReference"

688

Assertion ID	JCA-TA-9035
Source	[JCA80033]
Target	invocation of ComponentContext.cast method
Prerequisites	supplied target parameter is not an SCA reference proxy object
Predicate	method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "cast" "IllegalArgumentException"

689

Assertion ID	JCA-TA-9036

Source	[JCA80034]
Target	invocation of RequestContext.getSecuritySubject method
Prerequisites	current request has a JAAS subject method is invoked from code processing a service request or a callback request
Predicate	method returns the JAAS subject of the request
Prescription Level	mandatory
Tags	"RequestContext" "getSecuritySubject" "JAAS subject"

690

Assertion ID	JCA-TA-9037
Source	[JCA80034]
Target	invocation of RequestContext.getSecuritySubject method
Prerequisites	current request has no JAAS subject
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getSecuritySubject" "JAAS subject" "null"

691

Assertion ID	JCA-TA-9038
Source	[JCA80034]
Target	invocation of RequestContext.getSecuritySubject method
Prerequisites	method is invoked from code not processing a service request or a callback request
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getSecuritySubject" "JAAS subject" "null"

692

Assertion ID	JCA-TA-9039
Source	[JCA80035]
Target	invocation of RequestContext.getServiceName method
Prerequisites	invocation is from thread processing service operation or callback operation
Predicate	method returns the name of the service for which operation is being processed

Prescription Level	mandatory
Tags	"RequestContext" "getServiceName" "operation"

693

Assertion ID	JCA-TA-9040
Source	[JCA80035]
Target	invocation of RequestContext.getServiceName method
Prerequisites	invocation is not from thread processing service operation or callback operation
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getServiceName" "null"

694

Assertion ID	JCA-TA-9041
Source	[JCA80036]
Target	invocation of RequestContext.getCallbackReference method
Prerequisites	method invoked from a thread processing an operation of a bidirectional service
Predicate	method returns a ServiceReference typed by the interface of the callback
Prescription Level	mandatory
Tags	"RequestContext" "getCallbackReference" "bidirectional service"

695

Assertion ID	JCA-TA-9042
Source	[JCA80036]
Target	invocation of RequestContext.getCallbackReference method
Prerequisites	method invoked from a thread processing an operation of a service which is not bidirectional
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getCallbackReference" "not bidirectional service"

696

Assertion ID	JCA-TA-9043
Source	[JCA80036]

Target	invocation of RequestContext.getCallbackReference method
Prerequisites	method invoked from a thread not processing an operation of a service
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getCallbackReference" "null service"

697

Assertion ID	JCA-TA-9044
Source	[JCA80037]
Target	invocation of RequestContext.getCallback method
Prerequisites	method invoked from a thread processing an operation of a bidirectional service
Predicate	method returns a reference proxy object typed by the interface of the call-back
Prescription Level	mandatory
Tags	"RequestContext" "getCallback" "bidirectional service"

698

Assertion ID	JCA-TA-9045
Source	[JCA80037]
Target	invocation of RequestContext.getCallback method
Prerequisites	method invoked from a thread processing an operation of a service which is not bidirectional
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getCallback" "not bidirectional service"

699

Assertion ID	JCA-TA-9046
Source	[JCA80037]
Target	invocation of RequestContext.getCallback method
Prerequisites	method invoked from a thread not processing an operation of a service
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getCallback" "null service"

700

Assertion ID	JCA-TA-9047
Source	[JCA80038]
Target	invocation of RequestContext.getServiceReference method
Prerequisites	method is invoked by a thread processing a callback operation
Predicate	method returns a ServiceReference that represents the callback service
Prescription Level	mandatory
Tags	"RequestContext" "getServiceReference" "callback operation" "Service-Reference"

701

Assertion ID	JCA-TA-9048
Source	[JCA80039]
Target	invocation of RequestContext.getServiceReference method
Prerequisites	method is invoked by a thread not involved in processing either a service operation or a callback operation
Predicate	method returns null
Prescription Level	mandatory
Tags	"RequestContext" "getServiceReference" "null"

702

Assertion ID	JCA-TA-9049
Source	[JCA80040]
Target	invocation of ServiceReference.getService method
Prerequisites	
Predicate	method returns a reference proxy object typed by the business interface of the reference and which can be used to invoke operations on the target service of the reference
Prescription Level	mandatory
Tags	"ServiceReference" "getService" "reference proxy"

703

Assertion ID	JCA-TA-9050
Source	[JCA80041]
Target	invocation of ServiceReference.getBusinessInterface method
Prerequisites	

Predicate	method returns a Class object representing the business interface of the reference
Prescription Level	mandatory
Tags	"ServiceReference" "getBusinessInterface" "reference" "interface"

704

Assertion ID	JCA-TA-9051
Source	[JCA80042]
Target	invocation of SCAClientFactory.newInstance(URI) method
Prerequisites	domainURI parameter identifies a valid SCA Domain
Predicate	method returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "SCAClientFactory"

705

Assertion ID	JCA-TA-9052
Source	[JCA80043]
Target	invocation of SCAClientFactory.newInstance(URI) method
Prerequisites	domainURI parameter does not identify a valid SCA Domain
Predicate	method throws a NoSuchDomainException
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "SCAClientFactory" "NoSuchDomainException"

706

Assertion ID	JCA-TA-9053
Source	[JCA80044]
Target	invocation of SCAClientFactory.newInstance(Properties, URI) method
Prerequisites	domainURI parameter identifies a valid SCA Domain
Predicate	method returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "Properties" "SCAClientFactory"

707

Assertion ID	JCA-TA-9054

Source	[JCA80045]
Target	invocation of SCAClientFactory.newInstance(Properties, URI) method
Prerequisites	domainURI parameter does not identify a valid SCA Domain
Predicate	method throws a NoSuchDomainException
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "Properties" "SCAClientFactory" "NoSuchDomainException"

708

Assertion ID	JCA-TA-9055
Source	[JCA80046]
Target	invocation of SCAClientFactory.newInstance(Classloader, URI) method
Prerequisites	domainURI parameter identifies a valid SCA Domain
Predicate	method returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "Classloader" "SCAClientFactory"

709

Assertion ID	JCA-TA-9056
Source	[JCA80047]
Target	invocation of SCAClientFactory.newInstance(Classloader, URI) method
Prerequisites	domainURI parameter does not identify a valid SCA Domain
Predicate	method throws a NoSuchDomainException
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "Classloader" "SCAClientFactory" "NoSuchDomainException"

710

Assertion ID	JCA-TA-9057
Source	[JCA80048]
Target	invocation of SCAClientFactory.newInstance(Properties, Classloader, URI) method
Prerequisites	domainURI parameter identifies a valid SCA Domain
Predicate	method returns an object which implements the SCAClientFactory class for the SCA Domain identified by the domainURI parameter

Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "Properties" "Classloader" "SCAClient-Factory"

711

Assertion ID	JCA-TA-9058
Source	[JCA80049]
Target	invocation of SCAClientFactory.newInstance(Properties, Classloader, URI) method
Prerequisites	domainURI parameter does not identify a valid SCA Domain
Predicate	method throws a NoSuchDomainException
Prescription Level	mandatory
Tags	"SCAClientFactory" "domainURI" "Properties" "Classloader" "SCAClient-Factory" "NoSuchDomainException"

712

Assertion ID	JCA-TA-9059
Source	[JCA80050]
Target	invocation of SCAClientFactory.getService method
Prerequisites	service identified by the serviceURI parameter exists service has a service interface compatible with the interface supplied inthe interfaze parameter
Predicate	method returns a proxy object which implements the business interface supplied by the interfaze parameter which can be used to invoke operations on the service identified by the serviceURI parameter
Prescription Level	mandatory
Tags	"SCAClientFactory" "getService" "proxy"

713

Assertion ID	JCA-TA-9060
Source	[JCA80051]
Target	invocation of SCAClientFactory.getService method
Prerequisites	service identified by the serviceURI parameter does not exist OR service does not have a service interface compatible with the interface supplied inthe interfaze parameter
Predicate	method throws a NoSuchServiceException

Prescription Level	mandatory
Tags	"SCAClientFactory" "getService" "NoSuchServiceException"

714

Assertion ID	JCA-TA-9061
Source	[JCA80052]
Target	invocation of SCAClientFactory.getService method
Prerequisites	domain URI associated with SCAClientFactory does not itentify a valid SCA Domain
Predicate	method throws a NoSuchServiceException
Prescription Level	mandatory
Tags	"SCAClientFactory" "getService" "NoSuchServiceException"

715

Assertion ID	JCA-TA-9062
Source	[JCA80053]
Target	invocation of SCAClientFactory.getDomainURI method
Prerequisites	
Predicate	returns the SCA Domain URI of the Domain associated with the SCAClientFactory object
Prescription Level	mandatory
Tags	"SCAClientFactory" "getDomainURI" "Domain URI"

716

Assertion ID	JCA-TA-9063
Source	[JCA80054]
Target	invocation of SCAClientFactory.getDomainURI method
Prerequisites	domain URI associated with SCAClientFactory does not itentify a valid SCA Domain
Predicate	method throws a NoSuchServiceException
Prescription Level	mandatory
Tags	"SCAClientFactory" "getDomainURI" "NoSuchServiceException"

717

Assertion ID	JCA-TA-9064
Source	[JCA80055]

Target	invocation of SCAClientFactoryFinder.find method
Prerequisites	SCAClientFactory implementation exists and can be found
Predicate	returns an object which implements the SCAClientFactory interface for the SCA Domain represented by the domainURI parameter
Prescription Level	mandatory
Tags	"SCAClientFactoryFinder" "find" "SCAClientFactory"

718

Assertion ID	JCA-TA-9065
Source	[JCA80056]
Target	invocation of SCAClientFactoryFinder.find method
Prerequisites	SCAClientFactory implementation cannot be found
Predicate	method throws a ServiceRuntimeException
Prescription Level	mandatory
Tags	"SCAClientFactoryFinder" "find" "ServiceRuntimeException"

719

Assertion ID	JCA-TA-9066
Source	[JCA80057]
Target	invocation of ResponseDispatch.sendResponse() method
Prerequisites	sendResponse() and sendFault() methods of the ResponseDispatch object have not been called previously
Predicate	method sends the supplied response message to the client of the asyn-chronous service
Prescription Level	mandatory
Tags	"ResponseDispatch" "sendResponse"

720

Assertion ID	JCA-TA-9067
Source	[JCA80058]
Target	invocation of ResponseDispatch.sendResponse() method
Prerequisites	one of sendResponse() or sendFault() methods of the ResponseDispatch object have been called previously
Predicate	method throws an InvalidStateException
Prescription Level	mandatory

Tags	"ResponseDispatch" "sendResponse" "InvalidStateException"

721

Assertion ID	JCA-TA-9068
Source	[JCA80059]
Target	invocation of ResponseDispatch.sendFault() method
Prerequisites	sendResponse() and sendFault() methods of the ResponseDispatch object have not been called previously
Predicate	method sends the supplied fault to the client of the asynchronous service
Prescription Level	mandatory
Tags	"ResponseDispatch" "sendFault"

722

Assertion ID	JCA-TA-9069
Source	[JCA80060]
Target	invocation of ResponseDispatch.sendFault() method
Prerequisites	one of sendResponse() or sendFault() methods of the ResponseDispatch object have been called previously
Predicate	method throws an InvalidStateException
Prescription Level	mandatory
Tags	"ResponseDispatch" "sendFault" "InvalidStateException"

723

Assertion ID	JCA-TA-9070
Source	[JCA80004]
Target	invocation of ComponentContext.getServiceReference() method
Prerequisites	reference named by referenceName parameter has a multiplicity > 1
Predicate	getServiceReference() method throws an IllegalArgumentException
Prescription Level	mandatory
Tags	"ComponentContext" "getServiceReference()" "multiplicity" "IllegalArgumentException"

724

725 # Appendix A.8 Test Assertions for SCA Java CAA Specification Section 10
726

727

Assertion ID	JCA-TA-10001

Source	[JCA90001]
Target	Component implementation contains an SCA annotation that is improperly used
Prerequisites	
Predicate	The component which uses the implementation does not run
Prescription Level	mandatory
Tags	"implementation" "invalid annotation"

728

Assertion ID	JCA-TA-10002
Source	[JCA90002]
Target	Component implementation class
Prerequisites	Class has static method Class has an SCA annotation
Predicate	Class does not have SCA annotation applied to static method
Prescription Level	mandatory
Tags	"SCA annotation" "static method"

729

730

Assertion ID	JCA-TA-10003
Source	[JCA90002]
Target	Component implementation class
Prerequisites	Class has static field Class has an SCA annotation
Predicate	Class does not have SCA annotation applied to static field
Prescription Level	mandatory
Tags	"SCA annotation" "static field"

731

732

Assertion ID	JCA-TA-10004
Source	[JCA90052]
Target	Java class containing @AllowsPassByReference annotation

Prerequisites	
Predicate	@AllowsPassByRefereence annotation is attached to: - the Java class itself - or a method of a remotable service offered by the class - or an SCA reference of the class, in which case it is attached to the reference in a place where a @Reference annotation is valid
Prescription Level	mandatory
Tags	"@AllowsPassByReference" "Java class" "attachment"

733

Assertion ID	JCA-TA-10031
Source	[JCA90046]
Target	@Callback annotation on a setter method or on a field of an implementation class, for injection of a callback object
Prerequisites	
Predicate	@Callback annotation does not specify any attributes
Prescription Level	mandatory
Tags	"annotation" "Callback" "implementation" "injection"

734

Assertion ID	JCA-TA-10005
Source	[JCA90003]
Target	Constructor of an implementation class
Prerequisites	Constructor has parameters Constructor is annotated with the @Constructor annotation
Predicate	Each parameter of the Constructor is annotated with either @Reference or with @Property
Prescription Level	mandatory
Tags	"annotation" "constructor" "parameters" "@Reference" "@Property"

735

Assertion ID	JCA-TA-10006
Source	[JCA90004]
Target	Method of Java implementation class annotated with @Destroy
Prerequisites	
Predicate	Method has void return type and no arguments

Prescription Level	mandatory
Tags	"annotation" "@Destroy" "return type" "arguments"

736

Assertion ID	JCA-TA-10007
Source	[JCA90005]
Target	Java implementation class with a method annotated with @Destroy
Prerequisites	Scope of the implementation class ends
Predicate	Method annotated with @Destroy is invoked.
Prescription Level	mandatory
Tags	"annotation" "@Destroy" "invoke" "scope"

737

Assertion ID	JCA-TA-10008
Source	[JCA90007]
Target	Java implementation class annotated with @EagerInit
Prerequisites	Implementation class is Composite scoped Component using the implementation is started
Predicate	Java implementation class is started
Prescription Level	mandatory
Tags	"annotation" "@EagerInit" "composite scope" "started"

738

Assertion ID	JCA-TA-10009
Source	[JCA90008]
Target	Method annotated with @Init in a Java implementation class
Prerequisites	
Predicate	Method has void return type and no arguments
Prescription Level	mandatory
Tags	"anotation" "method" "@Init" "return type" "arguments"

739

Assertion ID	JCA-TA-10010
Source	[JCA90009]
Target	Method annotated with @Init

Prerequisites	Java class is started Property and Reference injection is complete
Predicate	Method is invoked
Prescription Level	mandatory
Tags	"annotation" "@Init" "started" "invoked"

740

Assertion ID	JCA-TA-10011
Source	[JCA90011]
Target	Java implementation class field annotated with @Property
Prerequisites	
Predicate	Class field is not declared as final
Prescription Level	mandatory
Tags	"annotation" "@Property" "field" "final"

741

Assertion ID	JCA-TA-10012
Source	[JCA90013]
Target	@Property annotation of a constructor parameter
Prerequisites	
Predicate	@Property annotation has a @name attribute present
Prescription Level	mandatory
Tags	"annotation" "@Property" "@name" "constructor" "parameter"

742

743

Assertion ID	JCA-TA-10013
Source	[JCA90014]
Target	@Property annotation of a constructor parameter
Prerequisites	
Predicate	@Property annotation has @required=true
Prescription Level	mandatory
Tags	"annotation" "@Property" "@required" "constructor" "parameter"

744

Assertion ID	JCA-TA-10014
Source	[JCA90047]
Target	Introspected component type of Java implementation class
Prerequisites	@Property annnotation is applied to a class field or to the input parameter of a setter method or of a constructor Type of the element to which @Property is applied is an array or is a type that implements or extends java.util.Collection
Predicate	Component type contains a <property/> element corresponding to the @Property annotation, with @many attribute set to "true"
Prescription Level	mandatory
Tags	"annotation" "@Property" "componentType" "@many" "array" "collection"

745

Assertion ID	JCA-TA-10015
Source	[JCA90047]
Target	Introspected component type of Java implementation class
Prerequisites	@Property annnotation is applied to a class field or to the input parameter of a setter method or of a constructor Type of the element to which @Property is applied is not an array and is not a type that implements or extends java.util.Collection
Predicate	Component type contains a <property/> element corresponding to the @Property annotation, with @many attribute set to "false"
Prescription Level	mandatory
Tags	"annotation" "@Property" "componentType" "@many" "array" "collection"

746

Assertion ID	JCA-TA-10016
Source	[JCA90015]
Target	Definition of an Intent annotation
Prerequisites	The SCA intent has qualifiers
Predicate	The Intent annotation definition contains the @Qualifier annotation
Prescription Level	mandatory
Tags	"annotation" "defintion" "intent" "qualifiiers" "@Qualifier"

747

Assertion ID	JCA-TA-10017
Source	[JCA90016]

Target	Java implementation class field annotated with @Reference
Prerequisites	
Predicate	Class field is not declared as final
Prescription Level	mandatory
Tags	"annotation" "@Reference" "field" "final"

748

Assertion ID	JCA-TA-10018
Source	[JCA90018]
Target	@Reference annotation of a constructor method parameter
Prerequisites	
Predicate	@Reference annotation has a @name attribute present
Prescription Level	mandatory
Tags	"annotation" "constructor" "parameter" "@Reference" "@name"

749

Assertion ID	JCA-TA-10019
Source	[JCA0019]
Target	@Reference annotation of a constructor method parameter
Prerequisites	
Predicate	@Reference annotation has its @required attribute with the value "true"
Prescription Level	mandatory
Tags	"annotation" "constructor" "parameter" "@Reference" "@required"

750

Assertion ID	JCA-TA-10020
Source	[JCA90020]
Target	Introspected component type of Java implementation class
Prerequisites	@Reference annnotation is applied to a class field or to the input parameter of a setter method or of a constructor Type of the element to which @Reference is applied is not an array and is not a type that implements or extends java.util.Collection @Reference @required attribute has the value "false"
Predicate	Component type contains a <reference/> element corresponding to the @Refeence annotation, with @multiplicity attribute set to "0..1"
Prescription	mandatory

Level	
Tags	"annotation" "@Reference" "componentType" "@multiplicity" "array" "collection"

751

Assertion ID	JCA-TA-10021
Source	[JCA90020]
Target	Introspected component type of Java implementation class
Prerequisites	@Reference annnotation is applied to a class field or to the input parameter of a setter method or of a constructor Type of the element to which @Reference is applied is not an array and is not a type that implements or extends java.util.Collection @Reference @required attribute has the value "true"
Predicate	Component type contains a <reference/> element corresponding to the @Refeence annotation, with @multiplicity attribute set to "1..1"
Prescription Level	mandatory
Tags	"annotation" "@Reference" "componentType" "@multiplicity" "array" "collection"

752

Assertion ID	JCA-TA-10022
Source	[JCA90021]
Target	Introspected component type of Java implementation class
Prerequisites	@Reference annnotation is applied to a class field or to the input parameter of a setter method or of a constructor Type of the element to which @Reference is applied is an array or is a type that implements or extends java.util.Collection @Reference @required attribute has the value "false"
Predicate	Component type contains a <reference/> element corresponding to the @Refeence annotation, with @multiplicity attribute set to "0..n"
Prescription Level	mandatory
Tags	"annotation" "@Reference" "componentType" "@multiplicity" "array" "collection"

753

Assertion ID	JCA-TA-10023
Source	[JCA90021]
Target	Introspected component type of Java implementation class
Prerequisites	@Reference annnotation is applied to a class field or to the input para-

	meter of a setter method or of a constructor Type of the element to which @Reference is applied is an array or is a type that implements or extends java.util.Collection @Reference @required attribute has the value "true"
Predicate	Component type contains a <reference/> element corresponding to the @Refeence annotation, with @multiplicity attribute set to "1..n"
Prescription Level	mandatory
Tags	"annotation" "@Reference" "componentType" "@multiplicity" "array" "collection"

754

Assertion ID	JCA-TA-10024
Source	[JCA90022]
Target	Reference obtained by the Java implementation either via injection or via the ComponentContext getService(...) method
Prerequisites	Reference has multiplicity of 0..1 Component reference is unwired
Predicate	Reference is null
Prescription Level	mandatory
Tags	"reference" "unwired" "null" "0..1" "multiplicity"

755

Assertion ID	JCA-TA-10025
Source	[JCA90023]
Target	Reference obtained by the Java implementation either via injection or via the ComponentContext getService(...) method
Prerequisites	Reference has multiplicity of 0..n Component reference is unwired
Predicate	Reference is an empty array/collection
Prescription Level	mandatory
Tags	"reference" "unwired" "empty" "0..n" "multiplicity"

756

Assertion ID	JCA-TA-10026
Source	[JCA90024]
Target	Component reference is rewired during lifetime of an implementation instance

Prerequisites	
Predicate	Reference object is reinjected into the implementation instance
Prescription Level	optional
Tags	"reinjection" "reference" "rewire" "implementation" "instance"

757

Assertion ID	JCA-TA-10027
Source	[JCA90025]
Target	Implementation class which has a reference reinjected due to wiring changes
Prerequisites	SCA Runtime supports reinjection.
Predicate	Implementation is not STATELESS scope
Prescription Level	mandatory
Tags	"reinjection" "reference" "implementation" "stateless" "scope"

758

Assertion ID	JCA-TA-10028
Source	[JCA90025]
Target	Implementation class which has a reference reinjected due to wiring changes
Prerequisites	
Predicate	Implementation does not use constructor based injection for that reference
Prescription Level	mandatory
Tags	"reinjection" "reference" "implementation" "constructor"

759

Assertion ID	JCA-TA-10029
Source	[JCA90026]
Target	Reference object of an implementation instance
Prerequisites	Reference wiring changes so that reference target is changed Reference is not reinjected
Predicate	Reference object continues to work as if the reference target is not changed
Prescription Level	mandatory
Tags	"reference" "rewire" "reinjection"

760

761

Assertion ID	JCA-TA-10030
Source	[JCA90027]
Target	Operation invocation on a reference of an implementation instance
Prerequisites	Target of the reference has been undeployed since the reference was injected
Predicate	Operation throws InvalidServiceException
Prescription Level	recommended
Tags	"reference" "target" "undeployed" "invocation" "InvalidServiceException"

762

Assertion ID	JCA-TA-10032
Source	[JCA90028]
Target	Operation invocation on a reference of an implementation instance
Prerequisites	Target of the reference has become unavailable for some reason
Predicate	Operation throws ServiceUnavailableException
Prescription Level	recommended
Tags	"reference" "target" "unavailable" "invocation" "ServiceUnavailableException"

763

Assertion ID	JCA-TA-10033
Source	[JCA90029]
Target	Operation invocation on a reference of an implementation instance
Prerequisites	Target of the reference has been changed since the reference was injected
Predicate	Operation either works or throws InvalidServiceException
Prescription Level	mandatory
Tags	"reference" "target" "changed" "invocation" "InvalidServiceException"

764

Assertion ID	JCA-TA-10034
Source	[JCA90030]
Target	ServiceReference obtained from a reference object via the ComponentContext.cast() method
Prerequisites	Reference is reinjected

Predicate	ServiceReference object continues to work as if the reference target were not changed
Prescription Level	mandatory
Tags	"ServiceReference" "reference" "reinjection" "cast(...)"

765

Assertion ID	JCA-TA-10035
Source	[JCA90031]
Target	Operation invocation on ServiceReference object
Prerequisites	Target of the ServiceReference is undeployed
Predicate	Operation throws an InvalidServiceException
Prescription Level	recommended
Tags	"ServiceReference" "target" "undeployed" "InvalidServiceException" "invocation"

766

Assertion ID	JCA-TA-10036
Source	[JCA90032]
Target	Operation invocation on ServiceReference object
Prerequisites	Target of the ServiceReference becomes unavailable
Predicate	Operation throws a ServiceUnavailableException
Prescription Level	recommended
Tags	"ServiceReference" "target" "unavailable" "ServiceUnavailableException" "invocation"

767

Assertion ID	JCA-TA-10037
Source	[JCA90033]
Target	Operation invocation on a ServiceReference object
Prerequisites	Target of the ServiceReference has been changed since the ServiceReference was obtained
Predicate	Operation either works or throws InvalidServiceException
Prescription Level	mandatory
Tags	"ServiceReference" "target" "changed" "InvalidServiceException" "invocation"

768

Assertion ID	JCA-TA-10038
Source	[JCA90034]
Target	Reference obtained from getService() method of ComponentContext
Prerequisites	Domain configuration has changed since the component was started Configuration changes affect the reference
Predicate	Reference configuration reflects the Domain configuration at the time the getService() method was called
Prescription Level	mandatory
Tags	"reference" "getService()" "domain" "configuration" "changed"

769

Assertion ID	JCA-TA-10039
Source	[JCA90034]
Target	ServiceReference obtained from getServiceReference() method of ComponentContext
Prerequisites	Domain configuration has changed since the component was started Configuration changes affect the reference
Predicate	ServiceReference configuration reflects the Domain configuration at the time the getServiceReference() method was called
Prescription Level	mandatory
Tags	"ServiceReference" "getServiceReference()" "domain" "configuration" "changed"

770

771

Assertion ID	JCA-TA-10056
Source	[JCA90035]
Target	Business method of a reference obtained through the component context getService() method or of a ServiceReference obtained through the component context getServiceReference() method
Prerequisites	Target of reference/ServiceReference has been undeployed or has become unavailable
Predicate	Invocation of the business method throws InvalidServiceException or ServiceUnavailableException
Prescription Level	mandatory
Tags	"reference" "getService()" "ServiceReference" "getServiceReference()" "target" "undeployed" "unavailable"

772

Assertion ID	JCA-TA-10040
Source	[JCA90036]
Target	Reference obtained from getService() method of ComponentContext
Prerequisites	Target service of the component reference has changed since the component was started
Predicate	Reference target is the new target for the component reference
Prescription Level	recommended
Tags	"reference" "getService()" "target" "changed"

773

Assertion ID	JCA-TA-10041
Source	[JCA90036]
Target	ServiceReference obtained from getServiceReference() method of ComponentContext
Prerequisites	Target service of the component reference has changed since the component was started
Predicate	ServiceReference target is the new target for the reference
Prescription Level	recommended
Tags	"ServiceReference" "getServiceReference()" "target" "changed"

774

Assertion ID	JCA-TA-10042
Source	[JCA90037]
Target	Array or Collection object for a reference of multiplicity 0..n or 1..n
Prerequisites	Changes occur to the wiring of the reference or to the target(s) of the wiring Reference reinjection is not allowed
Predicate	Contents of the Array / Collection do not change
Prescription Level	mandatory
Tags	"reference" "multiplicity" "0..n" "1..n" "wiring" "target" "change" "injection"

775

776

Assertion ID	JCA-TA-10043
Source	[JCA90038]
Target	Reference with multiplicity 0..n or 1..n with a setter method for injection

Prerequisites	Changes occur to the wiring of the reference or to the target(s) of the wiring Reference was originally injected with an array or a collection object Runtime supports reference reinjection.
Predicate	Setter method is invoked with an array / collection object containing the new targets of the component reference
Prescription Level	mandatory
Tags	"reference" "multiplicity" "0..n" "1..n" "wiring" "target" "change" "injection"

777

Assertion ID	JCA-TA-10044
Source	[JCA90039]
Target	Array or Collection object for a reference, where reinjection takes place
Prerequisites	Reference of multiplicity 0..n or 1..n Changes occur to the wiring of the reference or to the target(s) of the wiring Runtime supports reinjection
Predicate	Array or Collection object is not the same object that was originally injected for this reference
Prescription Level	mandatory
Tags	"reference" "multiplicity" "0..n" "1..n" "wiring" "target" "change" "injection"

778

Assertion ID	JCA-TA-10045
Source	[JCA90040]
Target	Interface of a service marked remotable by the @Remotable annotation
Prerequisites	
Predicate	The interface is translatable into a WSDL portType
Prescription Level	mandatory
Tags	"interface" "service" "remotable" "WSDL" "portType" "translatable"

779

Assertion ID	JCA-TA-10046
Source	[JCA90046]
Target	@Scope annotation
Prerequisites	
Predicate	Annotation is applied to a component implementation class

Prescription Level	mandatory
Tags	"annotation" "@Scope" "implementation class"

780

Assertion ID	JCA-TA-10047
Source	[JCA90042]
Target	Implementation class annotated with @Service annotation
Prerequisites	@Service annotation declares a set of services which implies a set of service interfaces
Predicate	Implementation class implements all of the methods of all the declared service interfaces
Prescription Level	mandatory
Tags	"annotation" "@Service" "methods" "interface" "implementation"

781

Assertion ID	JCA-TA-10050
Source	[JCA90050]
Target	@names attribute of @Service annotation
Prerequisites	
Predicate	Number of strings in the @names attribute array is the same as the number of elements in the @value attribute array
Prescription Level	mandatory
Tags	"annotation" "@Service" "@names" "@interfaces" "number"

782

Assertion ID	JCA-TA-10055
Source	[JCA90045]
Target	Implementation class declaring 2 or more services
Prerequisites	
Predicate	Each service has a distinct Java simple name
Prescription Level	mandatory
Tags	"annotation" "@Service" "service" "name"

783

Assertion ID	JCA-TA-10058
Source	[JCA90060]

Target	@names attribute array of a @Service annotation
Prerequisites	@names attribute array has more than 1 entry
Predicate	Each entry in the @names attribute array is a unique string
Prescription Level	mandatory
Tags	"annotation" "@Service" "service" "names"

784

Assertion ID	JCA-TA-10059
Source	[JCA90053]
Target	@Remotable annotation
Prerequisites	
Predicate	Annotation applies to an interface class, a Java implementation class, a field, a setter method or a constructor parameter
Prescription Level	mandatory
Tags	"annotation" "@Remotable" "interface" "implementation" "field" "setter method" "constructor"

785

Assertion ID	JCA-TA-10060
Source	[JCA90054]
Target	Field or method of an implementation class annotated with @Callback
Prerequisites	Implementation class offers at least one bidirectional service
Predicate	Type of the field or of the method is the callback interface of one of the bidirectional services offered by the class
Prescription Level	mandatory
Tags	"annotation" "@Callback" "implementation" "field" "method"

786

Assertion ID	JCA-TA-10061
Source	[JCA90055]
Target	Method annotated with @OneWay
Prerequisites	
Predicate	Method has void return type and does not delare checked exceptions
Prescription Level	mandatory
Tags	"annotation" "@OneWay" "void return" "exceptions"

787

Assertion ID	JCA-TA-10062
Source	[JCA90056]
Target	Method of Java interface annotated with @OneWay
Prerequisites	
Predicate	All invocations of the method are executed in a non-blocking fashion
Prescription Level	mandatory
Tags	"annotation" "@OneWay" "invocation" "non blocking"

788

Assertion ID	JCA-TA-10063
Source	[JCA90057]
Target	Setter method or field of an implementation class with COMPOSITE scope
Prerequisites	
Predicate	Setter method or field is not annotated with @Callback
Prescription Level	mandatory
Tags	"annotation" "@Callback" "setter method" "field" "COMPOSITE"

789

Assertion ID	JCA-TA-10064
Source	[JCA90058]
Target	Setter method or field of implementation class, annotated with @Callback
Prerequisites	Component is invoked via a bidirectional service Stter method or field has a type which corresponds to the callback interface of the bidectional service
Predicate	SCA runtime injects a callback reference proxy into the setter method or field, when the Java class is initialized
Prescription Level	mandatory
Tags	"annotation" "@Callback" "invocation" "setter method" "field" "callback reference proxy"

790

Assertion ID	JCA-TA-10065
Source	[JCA90061]
Target	Field, setter method or constructor parameter annotated with @Property
Prerequisites	Java type of the field, setter method or constructor parameter is a primitive

	type or is a JAXB annotated class Component configuration contains a value for the property corresponding to the field, setter method or constructor parameter
Predicate	Component property value is converted into an instance of the Java type as defined by the XML to Java mapping in the JAXB specification, with XML schema validation enabled
Prescription Level	mandatory
Tags	"annotation" "@Property" "value" "JAXB" "mapping"

Appendix A.9 Test Assertions for SCA Java CAA Specification Section 11

Assertion ID	JCA-TA-11001
Source	[JCA100001]
Target	Java interface which is mapped to WSDL by SCA runtime
Prerequisites	Java interface does not have a @WebService annotation
Predicate	Interface is mapped to WSDL as if it did have @WebService annotation
Prescription Level	mandatory
Tags	"interface" "Java" "WSDL" "mapping" "@WebService"

Assertion ID	JCA-TA-11002
Source	[JCA100002]
Target	Java interface containing @org.oasisopen.sca.annotation.OneWay annotation, mapped to WSDL by SCA runtime
Prerequisites	
Predicate	Interface is mapped to WSDL as if it contained a @javax.jws.OneWay annotation
Prescription Level	mandatory
Tags	"interface" "Java" "WSDL" "mapping" "@OneWay"

Assertion ID	JCA-TA-11003
Source	[JCA100003]
Target	WSDL interface mapped to a Java interface by SCA runtime

Prerequisites	
Predicate	Generated @WebService annotation in Java interface is taken to mean that the service interface is Remotable
Prescription Level	mandatory
Tags	"interface" "Java" "WSDL" "mapping" "@WebService" "remotable"

797

Assertion ID	JCA-TA-11004
Source	[JCA100004]
Target	Java interface used for service interface
Prerequisites	Interface contains JAXB 2.1 data types and annotations
Predicate	SCA runtime is able to map the interface to WSDL
Prescription Level	mandatory
Tags	"interface" "WSDL" "mapping" "JAXB" "2.1"

798

Assertion ID	JCA-TA-11005
Source	[JCA100005]
Target	Java interface used for service interface
Prerequisites	Interface contains SDO 2.1 data types and annotations
Predicate	SCA runtime is able to map the interface to WSDL
Prescription Level	optional
Tags	"interface" "WSDL" "mapping" "SDO" "2.1"

799

Assertion ID	JCA-TA-11006
Source	[JCA100006]
Target	Java interface used to declare SCA service interface in <interface.java/> element
Prerequisites	
Predicate	Java interface does not contain additional client side asynchronous polling and callback methods defined by JAX-WS
Prescription Level	mandatory
Tags	"interface" "service" "asynchronous" "polling" "callback" "methods"

800

Assertion ID	JCA-TA-11007
Source	[JCA100007]
Target	Java interface used to declare SCA reference interface in a Java implementation class
Prerequisites	Java interface contains the additional client side asynchronous polling and callback methods defined by JAX-WS
Predicate	Java interface is treated as valid and is used
Prescription Level	optional
Tags	"interface" "reference" "asynchronous" "polling" "callback" "methods"

801

Assertion ID	JCA-TA-11008
Source	[JCA100008]
Target	SCA reference interface in the component type of a Java implementation class
Prerequisites	Reference is declared in the implementation class using a Java interface which contains the additional client side asynchronous polling and callback methods defined by JAX-WS
Predicate	Reference interface in the component type does not contain the additional client side asynchronous polling and callback methods defined by JAX-WS
Prescription Level	mandatory
Tags	"interface" "reference" "componentType" "asynchronous" "polling" "callback" "methods"

802

Assertion ID	JCA-TA-11009
Source	[JCA100009]
Target	Java interface used to declare SCA reference interface in a Java implementation class
Prerequisites	Java interface contains the additional client side asynchronous polling and callback methods defined by JAX-WS
Predicate	The implementation class is able to use the client side asynchronous polling and callback methods, with the semantics as described in the JAX-WS 2.1 specification
Prescription Level	mandatory
Tags	"interface" "reference" "asynchronous" "polling" "callback" "methods"

803

Assertion ID	JCA-TA-11010

Source	[JCA100010]
Target	interface of a service
Prerequisites	interface contains SCA defined service-side asynchronous methods
Predicate	SCA runtime supports the interface
Prescription Level	mandatory
Tags	"interface" "servicer-side" "service" "asynchronous methods"

804

Assertion ID	JCA-TA-11011
Source	[JCA100012]
Target	Java interface or Java class annotated with @WebService
Prerequisites	
Predicate	SCA runtime treats the interface or class as if it was annotated with @Remotable
Prescription Level	mandatory
Tags	"interface" "class" "@WebService" "@Remotable"

805

Assertion ID	JCA-TA-11012
Source	[JCA100013]
Target	Java class annotated with @WebService
Prerequisites	@wsdlLocation attribute of the @WebService annotation is set, referencing a WSDL document
Predicate	The interface of the Java class is defined by the referenced WSDL definition
Prescription Level	mandatory
Tags	"interface" "@WebService" "wsdlLocation" "class"

806

Assertion ID	JCA-TA-11013
Source	[JCA100014]
Target	Java class annotated with @WebService
Prerequisites	@endpointInterface attribute of the @WebService annotation is set, referencing an interface
Predicate	The interface of the Java class is defined by the referenced interface

Prescription Level	mandatory
Tags	"interface" "@WebService" "endpointInterface" "class"

807

Assertion ID	JCA-TA-11014
Source	[JCA100015]
Target	Java class or interface containing @WebParam annotation
Prerequisites	@WebParam annotation has @header attribute set to "true"
Predicate	Java class or interface is treated as if has the SOAP intent applied
Prescription Level	mandatory
Tags	"interface" "@WebParam" "@header" "SOAP"

808

Assertion ID	JCA-TA-11015
Source	[JCA100016]
Target	Java class or interface containing @WebResult annotation
Prerequisites	@WebResult annotation has @header attribute set to "true"
Predicate	Java class or interface is treated as if has the SOAP intent applied
Prescription Level	mandatory
Tags	"interface" "@WebResult" "@header" "SOAP"

809

Assertion ID	JCA-TA-11016
Source	[JCA100017]
Target	Java class containing @ServiceMode annotation
Prerequisites	
Predicate	Java class is treated as if has the SOAP intent applied
Prescription Level	mandatory
Tags	"interface" "@ServiceMode" "SOAP"

810

Assertion ID	JCA-TA-11017
Source	[JCA100018]
Target	Java interface or class used to define an SCA interface
Prerequisites	

Predicate	Interface or class does not contain a @WebServiceClient annotation
Prescription Level	mandatory
Tags	"interface" "class" "@WebServiceClient"

811

Assertion ID	JCA-TA-11018
Source	[JCA100019]
Target	Java class annotated with @WebServiceProvider
Prerequisites	
Predicate	Class is treated as if annotated with SCA @Remotable
Prescription Level	mandatory
Tags	"class" "WebServiceProvider" "@Remotable"

812

Assertion ID	JCA-TA-11019
Source	[JCA100020]
Target	Java class annotated with @WebServiceProvider
Prerequisites	@wsdlLocation attribute of @WebServiceProvider is declared, referencing a WSDL document
Predicate	Java class is treated as if its interface is defined by the referenced WSDL
Prescription Level	mandatory
Tags	"class" "interface" "@WebServiceProvider" "@wsdlLocation"

813

Assertion ID	JCA-TA-11020
Source	[JCA100021]
Target	Java class or interface annotated with @SOAPBinding
Prerequisites	
Predicate	Class or interface is treated as if SOAP intent is applied
Prescription Level	mandatory
Tags	"interface" "class" "@SOAPBinding" "SOAP"

814

Assertion ID	JCA-TA-11021
Source	[JCA100022]

Target	Mappings from WSDL to Java and from Java to WSDL
Prerequisites	
Predicate	JAX-WS 2.1 mappings are supported
Prescription Level	mandatory
Tags	"interface"

815

Assertion ID	JCA-TA-11022
Source	[JCA100023]
Target	Java class or Java interface annotated with @WebService
Prerequisites	@name attribute of the @WebService annotation is set and the Java class or Java interface is used to define a service for an implementation class that is not annotated with @Service
Predicate	The name of the service is the value of the @name attribute
Prescription Level	mandatory
Tags	"interface" "@WebService" "name" "class"

816

Assertion ID	JCA-TA-11023
Source	[JCA100024]
Target	Java method annotated with @WebMethod
Prerequisites	@operationName attribute of the @WebMethod annotation is set
Predicate	The name of the SCA operation corresponding to the Java method is the value of the @operationName attribute
Prescription Level	mandatory
Tags	"method" "@WebMethod" "operationName"

817

Assertion ID	JCA-TA-11024
Source	[JCA100025]
Target	Java method annotated with @WebMethod
Prerequisites	@exclude attribute of the @WebMethod annotation is set to “true”
Predicate	There is no SCA operation corresponding to the Java method in the inter-face
Prescription Level	mandatory

Tags	"interface" "@WebMethod" "exclude"

818

Assertion ID	JCA-TA-11025
Source	[JCA100026]
Target	Java parameter annotated with @WebMethod
Prerequisites	@mode attribute of the @WebParam annotation is set
Predicate	The directionality of the parameter matches the value of the @mode attribute
Prescription Level	mandatory
Tags	"parameter" "@WebParam" "mode"

819

Assertion ID	JCA-TA-11026
Source	[JCA100028]
Target	SCA service identified by both a @Service annotation and a @WebService annotation
Prerequisites	Java class or Java interface annotated with @WebService is used to define a service for an implementation class that is annotated with @Service with a names attribute.
Predicate	The value for the name of the service from the @Service attribute is used in the componentType
Prescription Level	mandatory
Tags	"interface" “@Service” "@WebService" "name" "class"

820

Appendix B Cross Mapping of Normative Statements to Test Assertions

Conformance statement	Test Assertion
JCA20001	JCA-TA-2001
JCA20002	JCA-TA-2002
JCA20003	JCA-TA-2003
JCA20004	JCA-TA-2004
JCA20005	JCA-TA-2005
JCA20006	JCA-TA-2006
JCA20007	JCA-TA-2007
JCA20008	JCA-TA-2008
JCA20009	JCA-TA-2009
JCA20010	JCA-TA-2010 JCA-TA-2011

Conformance statement	Test Assertion
JCA30001	JCA-TA-3001
JCA30002	JCA-TA-3002
JCA30003	JCA-TA-3003
JCA30004	JCA-TA-3004
JCA30005	JCA-TA-3005
JCA30006	JCA-TA-3006
JCA30007	JCA-TA-3007
JCA30009	JCA-TA-3008
JCA30010	JCA-TA-3009

Conformance statement	Test Assertion
JCA40001	JCA-TA-4001
JCA40002	JCA-TA-4002
JCA40003	JCA-TA-4003
JCA40004	JCA-TA-4004
JCA40005	JCA-TA-4005
JCA40006	JCA-TA-4006
JCA40007	JCA-TA-4007
JCA40008	JCA-TA-4008
JCA40009	JCA-TA-4009
JCA40010	JCA-TA-4010
JCA40011	JCA-TA-4011
JCA40012	JCA-TA-4012
JCA40013	JCA-TA-4013

Conformance statement	Test Assertion
JCA40014	JCA-TA-4014
JCA40015	JCA-TA-4015
JCA40016	JCA-TA-4016
JCA40017	JCA-TA-4017
JCA40018	JCA-TA-4018
JCA40019	JCA-TA-4019
JCA40020	JCA-TA-4020
JCA40021	JCA-TA-4021
JCA40022	JCA-TA-4022
JCA40023	JCA-TA-4023
JCA40024	JCA-TA-4024

828

Conformance statement	Test Assertion
JCA60001	JCA-TA-7001
JCA60002	JCA-TA-7002
JCA60003	JCA-TA-7003 JCA-TA-7004
JCA60004	JCA-TA-7005
JCA60005	JCA-TA-7006
JCA60006	JCA-TA-7007

829

Conformance statement	Test Assertion
JCA70001	JCA-TA-8001
JCA70002	JCA-TA-8002 JCA-TA-8003 JCA-TA-8004
JCA70003	JCA-TA-8005
JCA70004	JCA-TA-8006
JCA70005	JCA-TA-8007 JCA-TA-8008 JCA-TA-8009
JCA70006	JCA-TA-8010

830

Conformance statement	Test Assertion
JCA80001	JCA-TA-9001
JCA80002	JCA-TA-9002 JCA-TA-9005
JCA80003	JCA-TA-9003 JCA-TA-9004
JCA80004	JCA-TA-9070
JCA80005	JCA-TA-9006
JCA80006	JCA-TA-9007

Conformance statement	Test Assertion
JCA80007	JCA-TA-9008
JCA80008	JCA-TA-9009
JCA80009	JCA-TA-9010
JCA80010	JCA-TA-9011
JCA80011	JCA-TA-9012
JCA80012	JCA-TA-9013
JCA80013	JCA-TA-9014
JCA80014	JCA-TA-9015
JCA80015	JCA-TA-9016
JCA80016	JCA-TA-9017
JCA80017	JCA-TA-9018
JCA80018	JCA-TA-9019
JCA80019	JCA-TA-9020
JCA80020	JCA-TA-9021
JCA80021	JCA-TA-9022
JCA80022	JCA-TA-9023
JCA80023	JCA-TA-9024
JCA80024	JCA-TA-9025
JCA80025	JCA-TA-9026
JCA80026	JCA-TA-9027
JCA80027	JCA-TA-9028
JCA80028	JCA-TA-9029
JCA80029	JCA-TA-9030 JCA-TA-9031
JCA80030	JCA-TA-9032
JCA80031	JCA-TA-9033
JCA80032	JCA-TA-9034
JCA80033	JCA-TA-9035
JCA80034	JCA-TA-9036 JCA-TA-9037 JCA-TA-9038

Conformance statement	Test Assertion
JCA80035	JCA-TA-9039 JCA-TA-9040
JCA80036	JCA-TA-9041 JCA-TA-9042 JCA-TA-9043
JCA80037	JCA-TA-9044 JCA-TA-9045 JCA-TA-9046
JCA80038	JCA-TA-9047
JCA80039	JCA-TA-9048
JCA80040	JCA-TA-9049
JCA80041	JCA-TA-9050
JCA80042	JCA-TA-9051
JCA80043	JCA-TA-9052
JCA80044	JCA-TA-9053
JCA80045	JCA-TA-9054
JCA80046	JCA-TA-9055
JCA80047	JCA-TA-9056
JCA80048	JCA-TA-9057
JCA80049	JCA-TA-9058
JCA80050	JCA-TA-9059
JCA80051	JCA-TA-9060
JCA80052	JCA-TA-9061
JCA80053	JCA-TA-9062
JCA80054	JCA-TA-9063
JCA80055	JCA-TA-9064
JCA80056	JCA-TA-9065
JCA80057	JCA-TA-9066
JCA80058	JCA-TA-9067
JCA80059	JCA-TA-9068
JCA80060	JCA-TA-9069

Conformance statement	Test Assertion
JCA90001	JCA-TA-10001
JCA90002	JCA-TA-10002 JCA-TA-10003
JCA90003	JCA-TA-10005
JCA90004	JCA-TA-10006
JCA90005	JCA-TA-10007
JCA90007	JCA-TA-10008
JCA90008	JCA-TA-10009
JCA90009	JCA-TA-10010
JCA90011	JCA-TA-10011
JCA90013	JCA-TA-10012
JCA90014	JCA-TA-10013
JCA90015	JCA-TA-10016
JCA90016	JCA-TA-10017
JCA90018	JCA-TA-10018
JCA90019	JCA-TA-10019
JCA90020	JCA-TA-10020 JCA-TA-10021
JCA90021	JCA-TA-10022 JCA-TA-10023
JCA90022	JCA-TA-10024
JCA90023	JCA-TA-10025
JCA90024	JCA-TA-10026
JCA90025	JCA-TA-10027 JCA-TA-10028
JCA90026	JCA-TA-10029
JCA90027	JCA-TA-10030
JCA90028	JCA-TA-10032
JCA90029	JCA-TA-10033
JCA90030	JCA-TA-10034
JCA90031	JCA-TA-10035

Conformance statement	Test Assertion
JCA90032	JCA-TA-10036
JCA90033	JCA-TA-10037
JCA90034	JCA-TA-10038 JCA-TA-10039
JCA90035	JCA-TA-10056
JCA90036	JCA-TA-10040 JCA-TA-10041
JCA90037	JCA-TA-10042
JCA90038	JCA-TA-10043
JCA90039	JCA-TA-10044
JCA90040	JCA-TA-10045
JCA90041	JCA-TA-10046
JCA90042	JCA-TA-10047
JCA90045	JCA-TA-10055
JCA90046	JCA-TA-10031
JCA90047	JCA-TA-10014 JCA-TA-10015
JCA90050	JCA-TA-10050
JCA90052	JCA-TA-10004
JCA90053	JCA-TA-10059
JCA90054	JCA-TA-10060
JCA90055	JCA-TA-10061
JCA90056	JCA-TA-10062
JCA90057	JCA-TA-10063
JCA90058	JCA-TA-10064

Conformance statement	Test Assertion
JCA90060	JCA-TA-10058
JCA90061	JCA-TA-10065

832

833

Conformance statement	Test Assertion
JCA10001	JCA-TA-11001
JCA10002	JCA-TA-11002
JCA10003	JCA-TA-11003
JCA10004	JCA-TA-11004
JCA10005	JCA-TA-11005
JCA10006	JCA-TA-11006
JCA10007	JCA-TA-11007
JCA10008	JCA-TA-11008
JCA10009	JCA-TA-11009
JCA10010	JCA-TA-11010
JCA10011	No meaningful assertion
JCA10012	JCA-TA-11011
JCA10013	JCA-TA-11012
JCA10014	JCA-TA-11013
JCA10015	JCA-TA-11014
JCA10016	JCA-TA-11015
JCA10017	JCA-TA-11016
JCA10018	JCA-TA-11017
JCA10019	JCA-TA-11018
JCA10020	JCA-TA-11019
JCA10021	JCA-TA-11020
JCA10022	JCA-TA-11021
JCA10023	JCA-TA-11022
JCA10024	JCA-TA-11023

Conformance statement	Test Assertion
JCA10025	JCA-TA-11024
JCA10026	JCA-TA-11025
JCA10028	JCA-TA-11026

834

Appendix C Cross Mapping of Test Assertions to TestCases

Test Assertion	Test Cases
ASM-TA-8001	JCA_2001_TestCase
ASM-TA-8005	JCA_1001_TestCase
ASM-TA-8006	JCA_1002_TestCase

Test Assertion	Test Cases
JCA-TA-2001	JCA_2001_TestCase
JCA-TA-2002	JCA_2002_TestCase
JCA-TA-2003	JCA_2003_TestCase
JCA-TA-2004	JCA_2004_TestCase
JCA-TA-2005	JCA_2005_TestCase
JCA-TA-2006	JCA_2006_TestCase
JCA-TA-2007	JCA_2007_TestCase
JCA-TA-2008	JCA_2008_TestCase
JCA-TA-2009	Optional - not tested
JCA-TA-2010	JCA_2009_TestCase
JCA-TA-2011	JCA_2010_TestCase

Test Assertion	Test Cases
JCA-TA-3001	JCA_3001_TestCase JCA_3002_TestCase
JCA-TA-3002	JCA_3003_TestCase JCA_3004_TestCase
JCA-TA-3003	JCA_3005_TestCase JCA_3006_TestCase JCA_3007_TestCase
JCA-TA-3004	JCA_3008_TestCase
JCA-TA-3005	JCA_3009_TestCase JCA_3010_TestCase

JCA-TA-3006	JCA_3011_TestCase
JCA-TA-3007	JCA_3012_TestCase
JCA-TA-3008	JCA_3013_TestCase
JCA-TA-3009	JCA_3014_TestCase

840

Test Assertion	Test Cases
JCA-TA-4001	JCA_4001_TestCase
JCA-TA-4002	JCA_4001_TestCase
JCA-TA-4003	JCA_4001_TestCase
JCA-TA-4004	JCA_4002_TestCase
JCA-TA-4005	JCA_4001_TestCase
JCA-TA-4006	JCA_4001_TestCase
JCA-TA-4007	JCA_4001_TestCase
JCA-TA-4008	JCA_4001_TestCase
JCA-TA-4009	JCA_4001_TestCase
JCA-TA-4010	JCA_4003_TestCase
JCA-TA-4011	JCA_4001_TestCase
JCA-TA-4012	JCA_4004_TestCase
JCA-TA-4013	JCA_4001_TestCase
JCA-TA-4014	JCA_4001_TestCase
JCA-TA-4015	JCA_4005_TestCase
JCA-TA-4016	JCA_4001_TestCase
JCA-TA-4017	JCA_4001_TestCase
JCA-TA-4018	JCA_4001_TestCase
JCA-TA-4019	untestable
JCA-TA-4020	JCA_4001_TestCase
JCA-TA-4021	JCA_4001_TestCase
JCA-TA-4022	JCA_4007_TestCase
JCA-TA-4023	JCA_4001_TestCase
JCA-TA-4024	JCA_4008_TestCase

841

842

Test Assertion	Test Cases
JCA-TA-7001	JCA_7001_TestCase
JCA-TA-7002	JCA_7002_TestCase
JCA-TA-7003	JCA_7003_TestCase JCA_7004_TestCase
JCA-TA-7004	JCA_7003_TestCase
JCA-TA-7005	JCA_7003_TestCase
JCA-TA-7006	JCA_7005_TestCase JCA_7006_TestCase
JCA-TA-7007	JCA_7003_TestCase

843

Test Assertion	Test Cases
JCA-TA-8001	JCA_8001_TestCase
JCA-TA-8002	JCA_8002_TestCase JCA_8005_TestCase
JCA-TA-8003	JCA_8003_TestCase JCA_8005_TestCase
JCA-TA-8004	JCA_8004_TestCase JCA_8005_TestCase
JCA-TA-8005	JCA_8006_TestCase
JCA-TA-8006	JCA_8007_TestCase
JCA-TA-8007	JCA_8008_TestCase
JCA-TA-8008	JCA_8009_TestCase
JCA-TA-8009	JCA_8010_TestCase
JCA-TA-8010	JCA_8011_TestCase

844

845

Test Assertion	Test Cases
JCA-TA-9001	JCA_9001_TestCase JCA_9008_TestCase
JCA-TA-9002	JCA_9002_TestCase
JCA-TA-9003	JCA_9003_TestCase

JCA-TA-9004	JCA_9004_TestCase
JCA-TA-9005	JCA_9005_TestCase
JCA-TA-9006	JCA_9006_TestCase
JCA-TA-9007	JCA_9006_TestCase
JCA-TA-9008	JCA_9006_TestCase
JCA-TA-9009	JCA_9007_TestCase
JCA-TA-9010	JCA_9008_TestCase
JCA-TA-9011	JCA_9008_TestCase
JCA-TA-9012	JCA_9008_TestCase
JCA-TA-9013	JCA_9008_TestCase
JCA-TA-9014	JCA_9006_TestCase
JCA-TA-9015	JCA_9009_TestCase
JCA-TA-9016	JCA_9009_TestCase
JCA-TA-9017	JCA_9009_TestCase
JCA-TA-9018	JCA_9009_TestCase
JCA-TA-9019	JCA_9009_TestCase
JCA-TA-9020	JCA_9010_TestCase
JCA-TA-9021	JCA_9010_TestCase
JCA-TA-9022	JCA_9010_TestCase
JCA-TA-9023	JCA_9010_TestCase
JCA-TA-9024	JCA_9010_TestCase
JCA-TA-9025	JCA_9011_TestCase
JCA-TA-9026	JCA_9011_TestCase
JCA-TA-9027	JCA_9012_TestCase
JCA-TA-9028	JCA_9012_TestCase
JCA-TA-9029	JCA_9012_TestCase
JCA-TA-9030	JCA_9013_TestCase
JCA-TA-9031	JCA_9013_TestCase
JCA-TA-9032	JCA_9013_TestCase
JCA-TA-9033	JCA_9013_TestCase

JCA-TA-9034	JCA_9014_TestCase
JCA-TA-9035	JCA_9014_TestCase
JCA-TA-9036	Untestable - no standard means of configuring a service to have a JAAS subject
JCA-TA-9037	Untestable - no standard means of configuring a service to have a JAAS subject
JCA-TA-9038	Untestable - no standard means of configuring a service to have a JAAS subject
JCA-TA-9039	JCA_9015_TestCase
JCA-TA-9040	JCA_9015_TestCase
JCA-TA-9041	JCA_9015_TestCase
JCA-TA-9042	JCA_9015_TestCase
JCA-TA-9043	JCA_9015_TestCase
JCA-TA-9044	JCA_9015_TestCase
JCA-TA-9045	JCA_9015_TestCase
JCA-TA-9046	JCA_9015_TestCase
JCA-TA-9047	JCA_9015_TestCase
JCA-TA-9048	JCA_9015_TestCase
JCA-TA-9049	JCA_9015_TestCase
JCA-TA-9050	JCA_9014_TestCase
JCA-TA-9051	JCA_9016_TestCase
JCA-TA-9052	JCA_9016_TestCase
JCA-TA-9053	JCA_9016_TestCase
JCA-TA-9054	JCA_9016_TestCase
JCA-TA-9055	JCA_9016_TestCase
JCA-TA-9056	JCA_9016_TestCase
JCA-TA-9057	JCA_9016_TestCase
JCA-TA-9058	JCA_9016_TestCase
JCA-TA-9059	JCA_9016_TestCase
JCA-TA-9060	JCA_9016_TestCase
JCA-TA-9061	Impossible to obtain a SCAClientFactory instance with an invalid domainURI - untestable

JCA-TA-9062	Untestable since this is a protected method.
JCA-TA-9063	Impossible to obtain a SCAClientFactory instance with an invalid domainURI - untestable
JCA-TA-9064	Untestable unless we make the vendor implementation of SCAClientFactoryFinder a conformance target
JCA-TA-9065	Untestable unless we make the vendor implementation of SCAClientFactoryFinder a conformance target
JCA-TA-9066	JCA_7003_TestCase
JCA-TA-9067	JCA_7005_TestCase
JCA-TA-9068	JCA_7006_TestCase
JCA-TA-9069	JCA_7006_TestCase
JCA-TA-9070	JCA_9006_TestCase

846

847

Test Assertion	Test Cases
JCA-TA-10001	JCA_10001_TestCase
JCA-TA-10002	JCA_10002_TestCase
JCA-TA-10003	JCA_10003_TestCase
JCA-TA-10004	JCA_2009_TestCase JCA_2010_TestCase
JCA-TA-10031	JCA_10004_TestCase
JCA-TA-10005	JCA_4001_TestCase JCA_10005_TestCase JCA_10006_TestCase
JCA-TA-10006	JCA_4001_TestCase JCA_10007_TestCase
JCA-TA-10007	JCA_4001_TestCase
JCA-TA-10008	JCA_2005_TestCase
JCA-TA-10009	JCA_10008_TestCase
JCA-TA-10010	JCA_4001_TestCase
JCA-TA-10011	JCA_10009_TestCase
JCA-TA-10012	JCA_10005_TestCase

	JCA_10010_TestCase
JCA-TA-10013	JCA_10005_TestCase JCA_10011_TestCase
JCA-TA-10014	JCA_10012_TestCase
JCA-TA-10015	JCA_2001_TestCase
JCA-TA-10016	JCA_10013_TestCase
JCA-TA-10017	JCA_2001_TestCase
JCA-TA-10018	JCA_10014_TestCase
JCA-TA-10019	JCA_10015_TestCase
JCA-TA-10020	JCA_2001_TestCase
JCA-TA-10021	JCA_2001_TestCase
JCA-TA-10022	JCA_10016_TestCase
JCA-TA-10023	JCA_2008_TestCase
JCA-TA-10024	JCA_10044_TestCase
JCA-TA-10025	JCA_10045_TestCase
JCA-TA-10026	optional requirement - no test
JCA-TA-10027	optional requirement - no test
JCA-TA-10028	optional requirement - no test
JCA-TA-10029	optional requirement - no test
JCA-TA-10030	requires undeploy API - no test
JCA-TA-10031	no TA
JCA-TA-10032	no TA
JCA-TA-10033	requires redeploy API - no test
JCA-TA-10034	optional requirement - no test
JCA-TA-10035	requires undeploy API - no test
JCA-TA-10036	requires undeploy API - no test
JCA-TA-10037	requires redeploy API - no test
JCA-TA-10038	requires deploy/redeploy API - no test
JCA-TA-10039	requires deploy/redeploy API - no test
JCA-TA-10040	requires deploy/redeploy API - no test
JCA-TA-10041	requires deploy/redeploy API - no test

JCA-TA-10042	requires deploy/redeploy API - no test
JCA-TA-10043	requires deploy/redeploy API - no test
JCA-TA-10044	requires optional support for rewiring - no test
JCA-TA-10045	JCA_10046_TestCase
JCA-TA-10046	JCA_10047_TestCase
JCA-TA-10047	JCA_10048_TestCase
JCA-TA-10050	JCA_10049_TestCase
JCA-TA-10055	JCA_10050_TestCase
JCA-TA-10056	requires deploy/redeploy API - no test
JCA-TA-10057	JCA_10051_TestCase
JCA-TA-10058	JCA_10052_TestCase
JCA-TA-10059	JCA_10029_TestCase JCA_10030_TestCase
JCA-TA-10060	JCA_10031_TestCase
JCA-TA-10061	JCA_10032_TestCase JCA_10033_TestCase
JCA-TA-10062	JCA_10034_TestCase
JCA-TA-10063	JCA_10035_TestCase
JCA-TA-10064	JCA_3005_TestCase
JCA-TA-10065	JCA_2006_TestCase

Test Assertion	Test Cases
JCA-TA-11001	JCA_11001_TestCase
JCA-TA-11002	JCA_11002_TestCase
JCA-TA-11003	JCA_11003_TestCase
JCA-TA-11004	JCA_11004_TestCase
JCA-TA-11005	JCA_11005_TestCase
JCA-TA-11006	JCA_11006_TestCase
JCA-TA-11007	optional - no test case
JCA-TA-11008	JCA_11007_TestCase

JCA-TA-11009	JCA_11008_TestCase
JCA-TA-11010	JCA_7003_TestCase JCA_7004_TestCase
JCA-TA-11011	JCA_11003_TestCase
JCA-TA-11012	JCA_3014_TestCase
JCA-TA-11013	JCA_11009_TestCase
JCA-TA-11014	JCA_11010_TestCase
JCA-TA-11015	JCA_11011_TestCase
JCA-TA-11016	JCA_11013_TestCase
JCA-TA-11017	JCA_11014_TestCase
JCA-TA-11018	JCA_11015_TestCase
JCA-TA-11019	JCA_11016_TestCase
JCA-TA-11020	JCA_11012_TestCase
JCA-TA-11021	Tested by a range of the testcases in this test suite - default mappings - mappings influenced by JAXWS annotations - JAXB type mapping are all dealt with in various different testcases, so there is no need to create a new specific testcase for this general requirement.
JCA-TA-11022	JCA_11018_TestCase
JCA-TA-11023	JCA_11019_TestCase
JCA-TA-11024	JCA_11020_TestCase
JCA-TA-11025	JCA_11021_TestCase
JCA-TA-11026	JCA_11022_TestCase

Appendix D Acknowledgments

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

Participants:

Participant Name	Affiliation
Bryan Aupperle	IBM
Vladislav Bezrukov	SAP AG*
David Booz	IBM
Martin Chapman	Oracle Corporation
Vamsavardhana Reddy Chillakuru	IBM
Mark Combellack	Avaya, Inc.
Mike Edwards	IBM
Anish Karmarkar	Oracle Corporation
Ashok Malhotra	Oracle Corporation
Plamen Pavlov	SAP AG*
Eric Wells	Hitachi, Ltd.

Appendix E Revision History

Revision	Date	Editor	Changes Made
1	09/25/09	Mike Edwards	Initial version
4	10/02/09	Dave Booz	Section 3 testcases
5	10/06/09	Dave Booz	Section 8, 8001-8005
7	10/07/09	Dave Booz	Complete Section 8
11	10/09/09	Mike Edwards	All sections complete
12	06/21/10	Mike Edwards	Testcases added for CD04 of Java CAA specification: JCA_4008 JCA_7001 - JCA_7003 JCA_9006 - JCA_9016
13	06/28/10	Mike Edwards	Further Testcases added for CD04: JCA_3013 - JCA_3014 JCA_7004 - JCA_7006 JCA11009 - JCA_11017
cd01	07/19/10	Mike Edwards	All changes accepted Frontmatter adjusted to OASIS requirements
cd01-rev1	10/19/10	Dave Booz	Applied Issues 202,212
cd01-rev2	11/01/10	Bryan Aupperle	Issue 215 Add missing expected output to several test cases
cd01-rev3	06/27/11	Mike Edwards	Issue 235 - merge TestAssertions document into this TestCases document, as an appendix
wd014	08/15/11	Mike Edwards	All changes accepted. Frontmatter and references updated for csd02 draft.
~~Revision~~	~~Date~~	~~Editor~~	~~Changes Made~~
~~1~~	~~09/25/09~~	~~Mike Edwards~~	~~Initial version~~
~~4~~	~~10/02/09~~	~~Dave Booz~~	~~Section 3 testcases~~
~~5~~	~~10/06/09~~	~~Dave Booz~~	~~Section 8, 8001-8005~~
~~7~~	~~10/07/09~~	~~Dave Booz~~	~~Complete Section 8~~
~~11~~	~~10/09/09~~	~~Mike Edwards~~	~~All sections complete~~
~~12~~	~~06/21/10~~	~~Mike Edwards~~	~~Testcases added for CD04 of Java CAA specification:~~ ~~JCA_4008~~ ~~JCA_7001 - JCA_7003~~ ~~JCA_9006 - JCA_9016~~

~~13~~	~~06/28/10~~	~~Mike Edwards~~	~~Further Testcases added for CD04:~~ ~~JCA_3013 - JCA_3014~~ ~~JCA_7004 - JCA_7006~~ ~~JCA11009 - JCA_11017~~
~~cd01~~	~~07/19/10~~	~~Mike Edwards~~	~~All changes accepted~~ ~~Frontmatter adjusted to OASIS requirements~~

857