TestCases for the SCA_J POJO Component Implementation Version 1.1 Specification

Working Draft 06

30 July 2010

Specification URIs:

This Version:

http://docs.oasis-open.org/opencsa/sca-j/sca-j-pojo-ci-1.1-testcases-wd06.html

http://docs.oasis-open.org/opencsa/sca-j/sca-j-pojo-ci-1.1-testcases-wd06.odt

http://docs.oasis-open.org/opencsa/sca-j/sca-j-pojo-ci-1.1-testcases-wd06.pdf(Authoritative)

Previous Version:

N/A

Latest Version:

http://docs.oasis-open.org/opencsa/sca-j/sca-j-pojo-ci-1.1-testcases.html

http://docs.oasis-open.org/opencsa/sca-j/sca-j-pojo-ci-1.1-testcases.odt

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

Technical Committee:

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

Chair(s):

Martin Chapman, Oracle
Mike Edwards, IBM

Editor(s):

Mike Edwards, IBM
Eric Wells, Hitachi
Plamen Pavlov, SAP

Related Work:

This document is related to:

- Service Component Architecture Java Common Annotations and APIs Specification Version 1.1

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/

Abstract:

This document defines the TestCases for the SCA 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 Assembly 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" 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/.

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

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

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

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

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

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

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

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

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

Table of Contents

1 Introduction

This document defines the TestCases for the SCA Assembly specification.

The tests described in this document are related to the Test Assertions described in the SCA POJO Component Implementation Test Assertions document [POJO-TA].

1.1 TestCase Structure

The SCA J POJO-CI 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 must be deployed and run
2. data indicating which service operation(s) 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.

The components in the composites are typically implemented by Java POJO classes.

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:

- POJO_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.
- POJO_General - a shared contribution containing implementation type independent artifacts that can be used by many testcases.

1.2 Namespaces and Java Package Names

The SCA POJO-CI 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.

[POJO-TA] SCA_J POJO ComponentImplementation Test Assertions

http://docs.oasis-open.org/sca-j/sca-j-pojo-ci-1.1-test-assertions_wd01.pdf

1.5 Non-normative References

[TBD] [TBD]

2 TestCases

2.1 Section 2

POJO_2001_TestCase

Testcase ID	POJO_2001_TestCase
Test Assertion	JCI-TA-2001
Description	Tests that a Java POJO implementation can define a service interface using a Java interface
Artifacts	POJO_2001_TestCase.java Test_POJO_2001.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1Impl.java
Expected output	Positive test: “POJO_2001 request service1 operation1 invoked”

POJO_2002_TestCase

Testcase ID	POJO_2002_TestCase
Test Assertion	JCI-TA-2001
Description	Tests that a Java POJO implementation can define a service interface using a Java class
Artifacts	POJO_2002_TestCase.java Test_POJO_2002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java serviceImplClass.java
Expected output	Positive test: “POJO_2002 request service1 operation1 invoked”

POJO_2003_TestCase

Testcase ID	POJO_2003_TestCase
Test Assertion	JCI-TA-2002
Description	Tests that a Java POJO implementation class implements all the operations defined by its service interface
Artifacts	POJO_2003_TestCase.java Test_POJO_2003.composite TestInvocation.wsdl TestClient_0002.composite MultiOperationService1.java incompleteOperationsImpl.java
Expected output	Negative test: "exception"

2.2 Section 5

POJO_5001_TestCase

Testcase ID	POJO_5001_TestCase
Test Assertion	JCI-TA-5001
Description	Tests that a Java POJO implementation class has either a public or a protected constructor
Artifacts	POJO_5001_TestCase.java Test_POJO_5001.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1PrivateConstructorImpl.java
Expected output	Negative test: “exception”

POJO_5002_TestCase

Testcase ID	POJO_5002_TestCase
Test Assertion	JCI-TA-5002
Description	Tests that where a POJO implementation has a constructor annotated with @Constructor, that this constructor is invoked when the implementation class is instantiated
Artifacts	POJO_5002_TestCase.java Test_POJO_5002.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1AnnotatedConstructorImpl.java
Expected output	Positive test: “POJO_5002 request service1 operation1 invoked @Constructor was invoked”

POJO_5003_TestCase

Testcase ID	POJO_5003_TestCase
Test Assertion	JCI-TA-5003
Description	Tests that where a POJO implementation has no constructor annotated with @Constructor but has a constructor with all its parameters either annotated with @Property or annotated with @Reference, that this constructor is invoked when the implementation class is instantiated
Artifacts	POJO_5003_TestCase.java Test_POJO_5003.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1AnnotatedParameterConstructorImpl.java
Expected output	Positive test: “POJO_5003 request service1 operation1 invoked fullAnnotated

	constructor was invoked”

POJO_5004_TestCase

Testcase ID	POJO_5004_TestCase
Test Assertion	JCI-TA-5004
Description	Tests that where a POJO implementation has no constructor annotated with @Constructor and no constructor with all its parameters either annotated with @Property or annotated with @Reference, but it has a no-arg constructor, that the no-arg constructor is invoked when the implementation class is instantiated
Artifacts	POJO_5004_TestCase.java Test_POJO_5004.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1NoArgConstructorImpl.java
Expected output	Positive test: “POJO_5004 request service1 operation1 invoked no-arg constructor was invoked”

POJO_5005_TestCase

Testcase ID	POJO_5005_TestCase
Test Assertion	JCI-TA-5005
Description	Tests that where a POJO implementation has no constructor annotated with @Constructor and no constructor with all its parameters either annotated with @Property or annotated with @Reference and does not have a no-arg constructor, that no constructor is invoked and an error is thrown by the SCA runtime
Artifacts	POJO_5005_TestCase.java Test_POJO_5005.composite TestInvocation.wsdl TestClient_0002.composite

	ASM_0002_Client.java Service1.java service1NoValidConstructorsImpl.java
Expected output	Negative test: "exception"

POJO_5006_TestCase

Testcase ID	POJO_5006_TestCase
Test Assertion	JCI-TA-5006
Description	Tests that a POJO implementation class has no more than 1 constructor annotated with @Constructor
Artifacts	POJO_5006_TestCase.java Test_POJO_5006.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1MultipleConstructorAnnotationsImpl.java
Expected output	Negative test: "exception"

POJO_5007_TestCase

Testcase ID	POJO_5007_TestCase
Test Assertion	JCI-TA-5007
Description	Tests that a POJO implementation class containing no @Constructor annotations has no more than 1 constructor with a non-empty parameter list where all the parameters are annotated with either @Property or with @Reference
Artifacts	POJO_5007_TestCase.java Test_POJO_5007.composite TestInvocation.wsdl

	TestClient_0002.composite ASM_0002_Client.java Service1.java service1MultipleAnnotatedConstructorsImpl.java
Expected output	Negative test: "exception"

2.3 Section 8

POJO_6001_TestCase

Testcase ID	POJO_6001_TestCase
Test Assertion	JCA-TA-6001
Description	Tests that a POJO implementation annotated with @Scope("STATELESS") runs with the operational characteristics of a STATELESS scoped implementation.
Artifacts	POJO_6001_TestCase.java Test_POJO_6001.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl2java
Expected output	Positive test: "POJO_6001 request service1 operation1 invoked target stateless scope service invoked successfully"

Testcase ID	POJO_6002_TestCase
Test Assertion	Tests that a POJO implementation annotated with @Scope("COMPOSITE") runs with the operational characteristics of a COMPOSITE scoped implementation.
Description	Tests that
Artifacts	POJO_6002_TestCase.java Test_POJO_6002.composite TestInvocation.wsdl

	TestClient_0002.composite ASM_0002_Client.java Service1.java service1CoordinatorImpl.java service1CompositeImpl.java
Expected output	Positive test: "POJO_6002 request service1 operation1 invoked target composite scope service invoked successfully"

2.4 Section 8

POJO_8001_TestCase

Testcase ID	POJO_8001_TestCase
Test Assertion	JCI-TA-8001
Description	Tests that when a POJO implementation class has a @Service annotation with a single interface class in its value attribute and no name attribute, that its componentType has a <service/> subelement with a @name attribute which is the simple name of the Java interface class
Artifacts	POJO_8001_TestCase.java Test_POJO_8001.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1Superset.java service1SupersetImpl.java
Expected output	Positive test: "POJO_8001 request service1 operation1 invoked"

POJO_8002_TestCase

Testcase ID	POJO_8002_TestCase
Test Assertion	JCA-TA-8002
Description	Tests that when a POJO implementation class has a @Service annotation

	with a single interface class in its value attribute and no name attribute, that its componentType has a <service/> subelement with an <interface.java/> subelement with the @interface attribute set to the fully qualified name of the interface class in the @Service annotation
Artifacts	POJO_8002_TestCase.java Test_POJO_8002.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1Superset.java (1st copy in org.oasisopen.sca.test package) Service1Superset.java (2nd copy in default package) service1SupersetImpl.java
Expected output	Negative test: "exception"

POJO_8003_TestCase

Testcase ID	POJO_8003_TestCase
Test Assertion	JCI-TA-8003
Description	Tests that an SCA POJO implementation with an @Requires annotation on the class and a @Service annotation, has a <service/> element in its componentType which has a @requires attribute containing the set of intents contained in the @Requires annotation
Artifacts	POJO_8003_TestCase.java Test_POJO_8003.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java PropagatesTransaction.java service1WithIntentImpl.java
Expected output	Negative test: "exception"

POJO_8004_TestCase

Testcase ID	POJO_8004_TestCase
Test Assertion	JCI-TA-8004
Description	Tests that a POJO implementation with no @Service annotation that implements an interface where the interface class is annotated with @Remotable has a componentType with a <service/> subelement with @name set to the simple name of the interface class and with an <interface.java/> subelement that is set to the fully qualified name of the interface class
Artifacts	POJO_8004_TestCase.java Test_POJO_8004.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1NoServiceAnnotationImpl.java
Expected output	Positive test: "POJO_8004 request service1 operation1 invoked"

POJO_8005_TestCase

Testcase ID	POJO_8005_TestCase
Test Assertion	JCI-TA-8005
Description	Tests that where a POJO implementation class has no @Service annotations and does not implement an interface class which is annotated with @Remotable, that the componentType of the implementation has a single <service/> element with @name set to the simple name of the implementation class and an <interface.java/> sublement with @interface set to the fully qualified name of the implementation class
Artifacts	POJO_8005_TestCase.java Test_POJO_8005.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java plainClassImpl.java
Expected output	Positive test: "POJO_8005 request service1 operation1 invoked"

POJO_8006_TestCase

Testcase ID	POJO_8006_TestCase
Test Assertion	JCI-TA-8006
Description	Tests that where a POJO implementation class has a @Reference annotation with the @name parameter, that the componentType of the implementation has a <reference/> element with its @name attribute set to the value of the @name parameter of the @Reference annotation
Artifacts	POJO_8006_TestCase.java Test_POJO_8006.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java service1RefWithNameImpl.java
Expected output	Positive test: "POJO_8006 request service1 operation1 invoked service2 operation1 invoked"

POJO_8007_TestCase

Testcase ID	POJO_8007_TestCase
Test Assertion	JCI-TA-8008
Description	Tests that where a POJO implementation has a @Reference annotation with @required=false and the annotation annotates a field with an interface type, that the componentType of the implementation has a <reference/> element with @multiplicity set to "0..1"
Artifacts	POJO_8007_TestCase.java Test_POJO_8007.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl6.java
Expected output	Positive test: "POJO_8007 request service1 operation1 invoked reference is null"

POJO_8008_TestCase

Testcase ID	POJO_8008_TestCase
Test Assertion	JCI-TA-8009
Description	Tests that where a POJO implementation has a @Reference annotation which annotates a field which also has a @Requires annotation that the componentType of the implementation as a <reference/> element with a @requires attribute set to the value of the @Requires annotation
Artifacts	POJO_8008_TestCase.java Test_POJO_8008.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java service1WithRefIntentImpl.java
Expected output	Negative test: "exception"

POJO_8009_TestCase

Testcase ID	POJO_8009_TestCase
Test Assertion	JCI-TA-8010
Description	Tests that a POJO implementation with a @Property annotation which annotates a setter method and the setter method parameter has a type which is not an array or a collection type has a componentType <property/> element with @name set to the JavaBeans property name derived from the name of the setter method and has @type set to the JAXB mapping of the parameter of the setter method
Artifacts	POJO_8009_TestCase.java Test_POJO_8009.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1PropertySetterImpl.java

Expected output	Positive test: "POJO_8009 request service1 operation1 invoked 2002.09.13 09:00:00"

POJO_8010_TestCase

Testcase ID	POJO_8010_TestCase
Test Assertion	JCI-TA-8011
Description	Tests that a POJO implementation with no @Reference annotations and also with no @Property annotations and which has a public setter method which is not part of a service interface and which has a parameter typed by an interface which is annotated with @Remotable has a componentType with a <reference/> element with @name set to the Javabeans property name derived from the setter method name and @multiplicity of 1..1 and with an <interface.java/> subelement which references the fully qualified name of the interface class
Artifacts	POJO_8010_TestCase.java Test_POJO_8010.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java service1UnannotatedRefSetterImpl.java
Expected output	Positive test: "POJO_8010 request service1 operation1 invoked service2 operation1 invoked"

POJO_8011_TestCase

Testcase ID	POJO_8011_TestCase
Test Assertion	JCI-TA-8012
Description	Tests that a POJO implementation with no @Reference annotations and also with no @Property annotations and which has a public setter method which is not part of a service interface and which has a parameter which is not typed by an interface annotated with @Remotable or an array or a parameterized java.util.Collection has a componentType with a <property/> element with @name set to the Javabeans property name derived from the setter method name and @type set to the JAXB mapping of the type of the setter method and with @many set to "false"
Artifacts	POJO_8011_TestCase.java

	Test_POJO_8011.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1UnannotatedPropertySetterImpl.java
Expected output	Positive test: "POJO_8011 request service1 operation1 invoked"

POJO_8012_TestCase

Testcase ID	POJO_8012_TestCase
Test Assertion	JCI-TA-8013
Description	Tests that a POJO implementation with a @Requires attribute on the class that the componentType has an <implementation.java/> subelement with a @requires attribute with the same value as the @Requires attribute
Artifacts	POJO_8012_TestCase.java Test_POJO_8012.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1WithIntentImpl.java
Expected output	Negative test: "exception"

POJO_8013_TestCase

Testcase ID	POJO_8013_TestCase
Test Assertion	JCI-TA-8014
Description	Tests that a POJO implementation class with 2 setter methods annotated with @Property have unique JavaBeans property names for the methods
Artifacts	POJO_8013_TestCase.java Test_POJO_8013.composite

	TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1DupPropertySettersImpl.java
Expected output	Negative test: "exception"

POJO_8014_TestCase

Testcase ID	POJO_8014_TestCase
Test Assertion	JCI-TA-8015
Description	Tests that a POJO implementation class with 2 setter methods annotated with @Reference have unique JavaBeans property names for the methods
Artifacts	POJO_8014_TestCase.java Test_POJO_8014.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1DupReferenceSettersImpl.java
Expected output	Negative test: "exception"

POJO_8015_TestCase

Testcase ID	POJO_8015_TestCase
Test Assertion	JCI-TA-8007
Description	Tests that where a POJO implementation has a @Reference annotation with @required=true and the annotation annotates a field which is an array type, that the componentType of the implementation has a <reference/> element with @multiplicity set to "1..n"
Artifacts	POJO_8015_TestCase.java Test_POJO_8015.composite TestInvocation.wsdl

	TestClient_0002.composite ASM_0002_Client.java Service1.java Service1Impl.java Service1Impl3.java
Expected output	Positive test: "POJO_8015 request service1 operation1 invoked service2 operation1 invokedservice3 operation1 invoked"

POJO_8016_TestCase

Testcase ID	POJO_8016_TestCase
Test Assertion	JCI-TA-8016
Description	Tests that a POJO implementation class with @Service annotation and @Remotable annotation, where the @Service annotation references an interface which does not contain an @Remotable annotation has a componentType with a <service/> element with <interface.java/> set with @interface set to the fully qualified name of the interface class in the value attribute of the @Service annotation and the @remotable attribute set to "true"
Artifacts	POJO_8016_TestCase.java Test_POJO_8016.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1NoRemotable.java Service1NoRemotableImpl.java
Expected output	Positive test: "POJO_8016 request service1 operation1 invoked"

POJO_8017_TestCase

Testcase ID	POJO_8017_TestCase
Test Assertion	JCI-TA-8017
Description	Tests that a POJO implementation class with @Reference annotation on a

	setter method where the interface type of the method has no @Remotable annotation but the setter method has a @Remotable annotation, the implementation has a componentType with a <reference/> element with <interface.java/> set with @interface set to the fully qualified name of the interface class which types the parameter of the setter method and the @remotable attribute set to "true"
Artifacts	POJO_8017_TestCase.java Test_POJO_8017.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1NoRemotable.java Service1NoRemotableRefImpl.java
Expected output	Positive test: "POJO_8017 request service1 operation1 invoked service2 operation1 invoked"

POJO_8018_TestCase

Testcase ID	POJO_8018_TestCase
Test Assertion	JCI-TA-8018
Description	Tests that an implementation class with a setter method that takes an interface type parameter and is annotated with @Reference has a componentType with a <reference/> element with an <interface.java/> subelement with @interface set to the fully qualified name of the interface type
Artifacts	POJO_8018_TestCase.java Test_POJO_8018.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1ReferenceSetterImpl.java
Expected output	Positive test: "POJO_8018 request service1 operation1 invoked service2 operation1

	invoked"

POJO_8019_TestCase

Testcase ID	POJO_8019_TestCase
Test Assertion	JCI-TA-8019
Description	Tests that an implementation class with a constructor parameter which has an interface type and is annotated with @Reference has a componentType with a <reference/> element with an <interface.java/> subelement with @interface set to the fully qualified name of the interface type
Artifacts	POJO_8019_TestCase.java Test_POJO_8019.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1ReferenceConstructorImpl.java
Expected output	Positive test: "POJO_8019 request service1 operation1 invoked service2 operation1 invoked"

POJO_8020_TestCase

Testcase ID	POJO_8020_TestCase
Test Assertion	JCI-TA-8020
Description	Tests that an implementation class with a @Service annotation and a @PolicySets annotation with a policySet declared has a componentType with a <service/> element with the @policySets attribute containing the policySet from the @PolicySets annotation
Artifacts	POJO_8020_TestCase.java Test_POJO_8020.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java

	service1Impl.java Service1ServicePolicySetsImpl.java test:PolicySet3 in definitions.xml of contribution
Expected output	Positive test: "POJO_8020 request service1 operation1 invoked"

POJO_8021_TestCase

Testcase ID	POJO_8021_TestCase
Test Assertion	JCI-TA-8021
Description	Tests that an implementation class with a @Reference annotation and a @PolicySets annotation with a policySet declared has a componentType with a <reference/> element with the @policySets attribute containing the policySet from the @PolicySets annotation
Artifacts	POJO_8021_TestCase.java Test_POJO_8021.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1ReferencePolicySetsImpl.java
Expected output	Positive test: "POJO_8021 request service1 operation1 invoked service2 operation1 invoked"

POJO_8022_TestCase

Testcase ID	POJO_8022_TestCase
Test Assertion	JCI-TA-8022
Description	Tests that an implementation class with zero @Service, @Reference, @Property annotations and a @PolicySets annotation on the class with one policy set declared, has a componentType with a <service/> element with a @policySets attribute present containing the policy set declared in the @PolicySets annotation
Artifacts	POJO_8022_TestCase.java Test_POJO_8022.composite

	TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1UnannotatedWithPolicySetsImpl.java
Expected output	Positive test: "POJO_8022 request service1 operation1 invoked service2 operation1 invoked"

POJO_8023_TestCase

Testcase ID	POJO_8023_TestCase
Test Assertion	JCI-TA-8023
Description	Tests that an implementation class with zero @Service, @Reference, @Property annotations and a @PolicySets annotation on a public field, typed by an interface annotated with @Remotable, with one policy set declared, has a componentType with a <reference/> element with a @policySets attribute present containing the policy set declared in the @PolicySets annotation
Artifacts	POJO_8023_TestCase.java Test_POJO_8023.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1UnannotatedWithRefPolicySetsImpl.java
Expected output	Positive test: "POJO_8023 request service1 operation1 invoked service2 operation1 invoked"

POJO_8024_TestCase

Testcase ID	POJO_8024_TestCase
Test Assertion	JCI-TA-8024
Description	Tests that an implementation class has a @PolicySets annotation on the

	class with one policy set declared, has a componentType with a <implementation.java/> element with a @policySets attribute present containing the policy set declared in the @PolicySets annotation
Artifacts	POJO_8024_TestCase.java Test_POJO_8024.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1ReferencePolicySetsImpl.java
Expected output	Positive test: "POJO_8024 request service1 operation1 invoked service2 operation1 invoked"

POJO_8025_TestCase

Testcase ID	POJO_8025_TestCase
Test Assertion	JCI-TA-8025
Description	Tests that an implementation class with zero @Service, @Reference, @Property annotations, which has a public setter method, not part of a service interface, with a parameter type which is an array of interface classes annotated with @Remotable has a componentType with a <reference/> element with @name set to the JavaBeans property name from the setter method name, @multiplicity set to 1..n and an <interface.java/> subelement which references the fully qualified name of the interface class
Artifacts	POJO_8025_TestCase.java Test_POJO_8025.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1ReferencePolicySetsImpl.java
Expected output	Positive test: "POJO_8025 request service1 operation1 invoked service2 operation1 invoked"

POJO_8026_TestCase

Testcase ID	POJO_8026_TestCase
Test Assertion	JCI-TA-8026
Description	Tests that an implementation class with zero @Service, @Reference, @Property annotations, which has a public setter method, not part of a service interface, with a parameter type which is a java.util.Collection parameterized by an interface class annotated with @Remotable has a componentType with a <reference/> element with @name set to the JavaBeans property name from the setter method name, @multiplicity set to 1..n and an <interface.java/> subelement which references the fully qualified name of the interface class
Artifacts	POJO_8026_TestCase.java Test_POJO_8026.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1ReferencePolicySetsImpl.java
Expected output	Positive test: "POJO_8026 request service1 operation1 invoked service2 operation1 invoked"

POJO_8027_TestCase

Testcase ID	POJO_8027_TestCase
Test Assertion	JCI-TA-8027
Description	Tests that an implementation class with zero @Service, @Reference, @Property annotations, which has a public setter method, not part of a service interface, with a parameter type which is a java.util.Collection not parameterized by an interface class annotated with @Remotable has a componentType with a <property/> element with @name set to the JavaBeans property name from the setter method name, @type set to the JAXB mapping of the type of the setter parameter and @many set to "true"
Artifacts	POJO_8027_TestCase.java Test_POJO_8027.composite TestInvocation.wsdl

	TestClient_0002.composite ASM_0002_Client.java Service1.java service1Impl.java Service1UnannotatedManyPropertySetterImpl.java
Expected output	Positive test: "POJO_8027 request service1 operation1 invoked 2002.09.13 09:00:00 2010.10.28 23:05:13";

POJO_8028_TestCase

Testcase ID	POJO_8028_TestCase
Test Assertion	JCI-TA-8028
Description	Tests that an implementation class with a javax.jws.WebService annotation and no @Service annotation has a component type with a <service/> element with @name set to the name property on the @WebService annotation and an <interface.java/> subelement with the @interface attribute set to the fully qualified name of the implementation class
Artifacts	POJO_8028_TestCase.java Test_POJO_8028.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebServiceImpl.java
Expected output	Positive test: "POJO_8028 request service1 operation1 invoked";

POJO_8029_TestCase

Testcase ID	POJO_8029_TestCase
Test Assertion	JCI-TA-8029
Description	Tests that an implementation class with a javax.jws.WebService annotation and no @Service annotation, where the @EndpointInterface annotation property is set to the name of a Java interface class, has a component type with a <service/> element with @name set to the name property on the @WebService annotation and an <interface.java/> subelement with the

	@interface attribute set to the value of the EndpointInterface property
Artifacts	POJO_8029_TestCase.java Test_POJO_8029.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebServiceEPImpl.java
Expected output	Positive test: "POJO_8029 request service1 operation1 invoked";

POJO_8030_TestCase

Testcase ID	POJO_8030_TestCase
Test Assertion	JCI-TA-8030
Description	Tests that an implementation class with a javax.jws.WebService annotation and no @Service annotation, where the @WsdlLocation annotation property is set to a WSDL document, has a component type with a <service/> element with @name set to the name property on the @WebService annotation and an <interface.wsdl/> subelement with the @interface attribute set to the portType referenced by the value of the WsdlLocation property
Artifacts	POJO_8030_TestCase.java Test_POJO_8030.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebServiceWSDLImpl.java
Expected output	Negative test: "exception"

POJO_8031_TestCase

Testcase ID	POJO_8031_TestCase
Test Assertion	JCI-TA-8031

Description	Tests that an implementation class with a service method with javax.jws.WebParam annotation on a parameter with the @header property set to "true", has a component type with a <service/> element with @requires attribute containing the "sca:SOAP" intent
Artifacts	POJO_8031_TestCase.java Test_POJO_8031.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebParamImpl.java
Expected output	Negative test: "exception"

POJO_8032_TestCase

Testcase ID	POJO_8032_TestCase
Test Assertion	JCI-TA-8032
Description	Tests that an implementation class with a service method with javax.jws.WebResult annotation on a parameter with the @header property set to "true", has a component type with a <service/> element with @requires attribute containing the "sca:SOAP" intent
Artifacts	POJO_8032_TestCase.java Test_POJO_8032.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebResultImpl.java
Expected output	Negative test: "exception"

POJO_8033_TestCase

Testcase ID	POJO_8033_TestCase
Test Assertion	JCI-TA-8033

Description	Tests that an implementation class annotated with a javax.jws.soap.SOAPBinding annotation has a component type with a <service/> element with @requires attribute containing the "sca:SOAP" intent
Artifacts	POJO_8033_TestCase.java Test_POJO_8033.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebResultImpl.java
Expected output	Negative test: "exception"

POJO_8034_TestCase

Testcase ID	POJO_8034_TestCase
Test Assertion	JCI-TA-8034
Description	Tests that an implementation class with a javax.xml.ws.WebServiceProvider annotation, has a component type with a <service/> element with an <interface.java/> subelement set to the fully qualified name of the implementation class
Artifacts	POJO_8034_TestCase.java Test_POJO_8034.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebResultImpl.java
Expected output	Positive test: "POJO_8034 request service1 operation1 invoked";

POJO_8035_TestCase

Testcase ID	POJO_8035_TestCase
Test Assertion	JCI-TA-8035

Description	Tests that an implementation class with a javax.xml.ws.WebServiceProvider annotation which has a wsdlLocation property set, pointing to a WSDL document, has a component type with a <service/> element with an <interface.wsdl/> subelement set to the portType referenced by the value of the wsdlLocation property
Artifacts	POJO_8035_TestCase.java Test_POJO_8035.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebResultImpl.java
Expected output	Negative test: "exception"

POJO_8036_TestCase

Testcase ID	POJO_8036_TestCase
Test Assertion	JCI-TA-8036
Description	Tests that an implementation class annotated with any JAX-WS annotation has a component type with a <service/> element with a <binding.ws/> subelement with the @wsdlElement attribute set to point to a WSDL document that uses the SOAP/HTTP binding
Artifacts	POJO_8036_TestCase.java Test_POJO_8036.composite TestInvocation.wsdl TestClient_0002.composite ASM_0002_Client.java Service1.java Service1JWSWebResultImpl.java
Expected output	Positive test: "POJO_8036 request service1 operation1 invoked";

2.5 Section 9

POJO_9001_TestCase

Testcase ID	POJO_9001_TestCase
Test Assertion	JCI-TA-9001
Description	Tests that if an SCA <component/> element has an <implementation.java/> subelement, that the <implementation.java/> element conforms to the sca-implementation-java.xsd schema
Artifacts	POJO_9001_TestCase.java Test_POJO_9001.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1Impl.java
Expected output	Negative test: "exception"

POJO_9002_TestCase

Testcase ID	POJO_9002_TestCase
Test Assertion	JCI-TA-9002
Description	Tests that where a <component/> has an <implementation.java/> subelement with a @class attribute that has the fully qualified name of a Java class that is contained in the same contribution as the composite file containing the <component/>, that the Java class is found and instantiated
Artifacts	POJO_9002_TestCase.java Test_POJO_9002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java x.service1Impl.java
Expected output	Positive test: "POJO_9002 request service1 operation1 invoked"

POJO_9003_TestCase

Testcase ID	POJO_9003_TestCase
Test Assertion	JCI-TA-9003
Description	Tests that if an <implementation.java/> element has a @class attribute which references a Java POJO implementation class and that class is imported by the contribution containing the composite from a second contribution, the version of the class from the second contribution is loaded and used even where a version of the same class is available in the contribution containing the composite
Artifacts	POJO_9003_TestCase.java Test_POJO_9003.composite TestInvocation.wsdl TestClient_0002.composite Service1.java x.service1.java (2 copies #1 = contribution POJO_General; #2 = contribution POJO_9003)
Expected output	Positive test: "POJO_9003 request service1 operation1"

POJO_9004_TestCase

Testcase ID	POJO_9004_TestCase
Test Assertion	JCI-TA-9004
Description	Tests that for an <implementation.java/> element in a composite file that specifies the name of a Java POJO class in its @class attribute, where a Java class of that fully qualified name exists a) in the same contribution (A) as the composite file b) in two other contributions in the Domain (B & C), both of which export the namespace of the Java class and where contribution A imports the namespace with <import.java/> specifying @location referencing only one of the Contributions (C) that the Java class is loaded from Contribution C.
Artifacts	POJO_9004_TestCase.java Test_POJO_9004.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1ContributionImpl.java contribution POJO_9004

	contribution ContributionB contribution ContributionC
Expected output	Positive test: "POJO_9004 request service1 operation1 invoked contributionC"

2.6 Section 10

POJO_10001_TestCase

Testcase ID	POJO_10001_TestCase
Test Assertion	JCI-TA-10001
Description	Tests that where an sca-contribution.xml file contains multiple <import.java/> elements, that the value of the @package attribute of each element is unique across all the <import.java/> elements
Artifacts	POJO_10001_TestCase.java Test_POJO_10001.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.java service1Impl.java
Expected output	Negative test: "exception"

POJO_10002_TestCase

Testcase ID	POJO_10002_TestCase
Test Assertion	JCI-TA-10002
Description	Tests that where a contribution contains an SCA component that references classes from a Java package not contained in the contribution, and where the sca-contribution.xml contains an <import.java/> for that Java package, and there is a second contribution in the Domain that contains the classes of the package and which exports the package, that the SCA component can be instantiated and that it uses the Java classes from the second contribution.

Artifacts	POJO_10002_TestCase.java Test_POJO_10002.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1Impl.java
Expected output	Positive test: "POJO_10002 request service1 operation1 invoked"

POJO_10003_TestCase

Testcase ID	POJO_10003_TestCase
Test Assertion	JCI-TA-10003
Description	Tests that where a contribution A contains a component with an implementation from a Java package not found in contribution A and where the sca-contribution.xml has an <import.java/> for that package with a @location attribute pointing at a contribution B and contribution B both contains classes for that Java package and also exports that package, that the classes from contribution B satisfy the import from contribution A and that the component is instantiated.
Artifacts	POJO_10003_TestCase.java Test_POJO_10003.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1ContributionImpl.java <POJO_ContributionB>
Expected output	Positive test: "POJO_10003 request service1 operation1 invoked ContributionB"

POJO_10004_TestCase

Testcase ID	POJO_10004_TestCase
Test Assertion	JCI-TA-10004
Description	Tests that where a contribution A has references to Java artifacts from 2 different Java packages X and Y and where both of these packages are imported, and where package X is exported by a contribution B with a uses statement specifying a specific version of package Y, that import of Y in

	contribution A must import the same version of Y as specified in the uses attribute in contribution B.
Artifacts	POJO_10004_TestCase.java Test_POJO_10004.composite TestInvocation.wsdl TestClient_0002.composite Service1.java service1ContributionImpl.java <POJO_ContributionC> <POJO_ContributionD>
Expected output	Negative test: "exception"

POJO_10005_TestCase

Testcase ID	POJO_10005_TestCase
Test Assertion	JCI-TA-10005
Description	Tests that where a contribution has an sca-contribution.xml file with multiple <export.java/> elements, that the @package attribute value of each element is unique
Artifacts	POJO_10005_TestCase.java Test_POJO_10005.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.java service1Impl.java
Expected output	Negative test: "exception"

POJO_10006_TestCase

Testcase ID	POJO_10006_TestCase
Test Assertion	JCI-TA-10006
Description	Tests that where a contribution has an sca-contribution.xml file with an

	<export.java/> element which references a given Java package, that the files of that package are contained within the contribution
Artifacts	POJO_10006_TestCase.java Test_POJO_10006.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.java service1Impl.java
Expected output	Negative test: "exception"

POJO_10007_TestCase

Testcase ID	POJO_10007_TestCase
Test Assertion	JCI-TA-10007
Description	Tests that all Java classes loaded from a Contribution are loaded by a class loader that is unique to the contribution
Artifacts	POJO_10007_TestCase.java Test_POJO_10007.composite TestInvocation.wsdl TestClient_0002.composite TestComposite1.composite Service1.java x.service1ComplexClient.java x.serviceComplex1Impl.java y.ServiceComplex1.java z.ComplexClass.java <POJO_10007 contribution> <POJO_ContributionE contribution> <POJO_ContributionF contribution>
Expected output	Positive test: "POJO_10007 request service1 operation1 invoked service2 operation1 invoked Service interface classloader same as Impl class classloader ComplexClass classloader different from Impl class classloader"

POJO_10008_TestCase

Testcase ID	POJO_10008_TestCase
Test Assertion	JCI-TA-10008
Description	Tests that where a contribution A uses one or more Java classes that are imported from a contribution B, that contribution B's classloader is used to load the classes imported into contribution A
Artifacts	POJO_10008_TestCase.java Test_POJO_10008.composite TestComposite1.composite (contribution POJO_ContributionE) TestInvocation.wsdl TestClient_0002.composite Service1.java service1ComplexClient.java (contribution POJO_10008) x.serviceComplex1Impl.java y.ServiceComplex1.java z.ComplexClass.java <POJO_10008 contribution> <POJO_ContributionE contribution>
Expected output	Positive test: "POJO_10008 request service1 operation1 invoked classloaders set correctly"

POJO_10009_TestCase

Testcase ID	POJO_10009_TestCase
Test Assertion	JCI-TA-10009
Description	Tests that the thread context classloader of the thread used to invoke an operation of a Java POJO component implementation is the class loader of the contribution that contains the POJO implementation class.
Artifacts	POJO_10009_TestCase.java Test_POJO_10009.composite TestInvocation.wsdl TestClient_0002.composite

	TestComposite1.composite Service1.wsdl
Expected output	Positive test: "POJO_10009 request service1 operation1 invoked thread context classloader is correct"

3 Cross Mapping of Test Assertions to TestCases

Test Assertion	Test Cases
JCI-TA-2001	POJO_2001_TestCase
JCI-TA-2002	POJO_2002_TestCase
~~JCI-TA-2003~~	~~POJO_2003_TestCase~~

Test Assertion	Test Cases
JCI-TA-4001	POJO_4001_TestCase
JCI-TA-4002	POJO_4002_TestCase

Test Assertion	Test Cases
JCI-TA-5001	POJO_5001_TestCase
JCI-TA-5002	POJO_5002_TestCase
JCI-TA-5003	POJO_5003_TestCase
JCI-TA-5004	POJO_5004_TestCase
JCI-TA-5005	POJO_5005_TestCase
JCI-TA-5006	POJO_5006_TestCase
JCI-TA-5007	POJO_5007_TestCase

Test Assertion	Test Cases
JCI-TA-6001	POJO_6001_TestCase
JCI-TA-6002	POJO_6002_TestCase

Test Assertion	Test Cases
JCI-TA-8001	POJO_8001_TestCase
JCI-TA-8002	POJO_8002_TestCase
JCI-TA-8003	POJO_8003_TestCase
JCI-TA-8004	POJO_8004_TestCase
JCI-TA-8005	POJO_8005_TestCase
JCI-TA-8006	POJO_8006_TestCase
JCI-TA-8007	POJO_8015_TestCase
JCI-TA-8008	POJO_8007_TestCase

JCI-TA-8009	POJO_8008_TestCase
JCI-TA-8010	POJO_8009_TestCase
JCI-TA-8011	POJO_8010_TestCase
JCI-TA-8012	POJO_8011_TestCase
JCI-TA-8013	POJO_8012_TestCase
JCI-TA-8014	POJO_8013_TestCase
JCI-TA-8015	POJO_8014_TestCase
JCI-TA-8016	POJO_8016_TestCase
JCI-TA-8017	POJO_8017_TestCase
JCI-TA-8018	POJO_8018_TestCase
JCI-TA-8019	POJO_8019_TestCase
JCI-TA-8020	POJO_8020_TestCase
JCI-TA-8021	POJO_8021_TestCase
JCI-TA-8022	POJO_8022_TestCase
JCI-TA-8023	POJO_8023_TestCase
JCI-TA-8024	POJO_8024_TestCase
JCI-TA-8025	POJO_8025_TestCase
JCI-TA-8026	POJO_8026_TestCase
JCI-TA-8027	POJO_8027_TestCase
JCI-TA-8028	POJO_8028_TestCase
JCI-TA-8029	POJO_8029_TestCase
JCI-TA-8030	POJO_8030_TestCase
JCI-TA-8031	POJO_8031_TestCase
JCI-TA-8032	POJO_8032_TestCase
JCI-TA-8033	POJO_8033_TestCase
JCI-TA-8034	POJO_8034_TestCase
JCI-TA-8035	POJO_8035_TestCase
JCI-TA-8036	POJO_8036_TestCase
JCI-TA-8037	Unimplementable

Test Assertion	Test Cases
JCI-TA-9001	POJO_9001_TestCase
JCI-TA-9002	POJO_9002_TestCase
JCI-TA-9003	POJO_9003_TestCase

JCI-TA-9004	POJO_9004_TestCase
JCI-TA-9005	untestable with current mandatory requirements
JCI-TA-9006	no obvious test available

Test Assertion	Test Cases
JCI-TA-10001	POJO_10001_TestCase
JCI-TA-10002	POJO_10002_TestCase
JCI-TA-10003	POJO_10003_TestCase
JCI-TA-10004	POJO_10004_TestCase
JCI-TA-10005	POJO_10005_TestCase
JCI-TA-10006	POJO_10006_TestCase
JCI-TA-10007	POJO_10007_TestCase
JCI-TA-10008	POJO_10008_TestCase
JCI-TA-10009	POJO_10009_TestCase

4 Catalog of Test Artifacts

4.1 Composite Files - lower level

Name	Valid	Description
TestClient_0002.composite		1 service offered to the Testrunner client 1 reference using Service1
TestComposite1.composite		1 service using Service1 interface 0 references
TestComposite2.composite		1 service using Service1 interface 0 references

4.2 Java Interfaces

Name	Description
Service1.java	Service1 interface
ServiceComplex1.java	Interface with 2 operations - 1 with Java class as input and String response - 1 with void input and String response

4.3 Java Implementation Classes

Name	Description
All classes in the org.oasisopen.sca.test package unless noted	
x.service1ComplexClientImpl.java <POJO_ContributionE>	1 service with Service1 interface 1 reference with ServiceComplex1 interface
x.serviceComplex1Impl <POJO_ContributionE>	1 service with ServiceComplex1 interface - operation1 checks the classloaders of the class, the interface class and the complex Java class - operation2 returns the hashCode of the classloader of the class
x.service1ContributionImpl.java <POJO_ContributionD>	1 service with Service1 interface - operation1 responds with a string ending in the contribution name
service1ContributionImpl.java <POJO_ContributionC>	1 service with Service1 interface - operation1 responds with a string ending in the contribution name
service1ContributionImpl.java <POJO_ContributionB>	1 service with Service1 interface - operation1 responds with a string ending in the contribution name
service1ContributionImpl.java <POJO_9005> <POJO_9004>	1 service with Service1 interface - operation1 responds with a string ending in the name = "ContributionA"
x.service1Impl.java <POJO_9003>	1 service with Service1 interface
service1DupReferenceSettersImpl.java <POJO_9001>	1 service with Service1 interface has a 2 setter methods annotated with @Reference and the Javabeans names of these two methods is the same

z.ComplexClass.java <POJO_ContributionF>	ComplexClass class - used in a service operation with a Java class as its input parameter

4.4 WSDL Interface Files

Name	Description

5 Conformance

There are no conformance statements relating to the TestCases.

Appendix A. 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 B. Revision History

Revision	Date	Editor	Changes Made
1	25/09/09	Mike Edwards	Initial version
4	02/10/09	Dave Booz	Section 3 testcases
5	06/10/09	Dave Booz	Section 8, 8001-8005
7	07/10/09	Dave Booz	Complete Section 8
WD01	12/11/09	Mike Edwards	All sections complete
wd02	18/11/09	Mike Edwards	Added testcases in 8xxx range
wd03	26/07/10	Mike Edwards	Corrected cross-mapping tables Added testcases POJO_8015, POJO_8016
wd04			
wd05	29/07/10	Mike Edwards	Added testcases: POJO_8027_TestCase POJO_8028_TestCase POJO_8029_TestCase POJO_8030_TestCase POJO_8031_TestCase POJO_8032_TestCase
wd06			