



# Service Component Architecture JMS Binding Specification Version 1.1

Committee Specification Draft [0506](#) /  
Public Review Draft [0304](#)

~~8 November 2010~~

14 July 2011

## Specification URIs:

### This ~~V~~version:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd04.pdf>  
(Authoritative)  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd04.html>  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd04.doc>

### Previous ~~V~~version:

<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd03.pdf>  
(Authoritative)  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd03.html>  
<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd03.doc>

### Latest ~~V~~version:

~~<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec.pdf> (Authoritative)~~  
~~<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec.html>~~  
~~<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec.doc>~~  
~~<http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec.pdf> (Authoritative)~~

## Technical Committee:

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

### Chair(s):

Simon Holdsworth ([simon\\_holdsworth@uk.ibm.com](mailto:simon_holdsworth@uk.ibm.com), ~~IBM~~), IBM

### Editor(s):

#### Editors:

Simon Holdsworth ([simon\\_holdsworth@uk.ibm.com](mailto:simon_holdsworth@uk.ibm.com), ~~IBM~~), IBM  
Anish Karmarkar ([Anish.Karmarkar@oracle.com](mailto:Anish.Karmarkar@oracle.com), ~~Oracle~~), Oracle

### Related ~~W~~work:

This specification replaces or supersedes:

- [Service Component Architecture JMS Binding Specification Version 1.00](#)

This specification is related to:

- [Service Component Architecture Assembly Model Specification Version 1.1](#)
- [SCA Policy Framework Version 1.1](#)

### Declared XML ~~Namespace(s)~~:namespace:

<http://docs.oasis-open.org/ns/opencsa/sca/200912>

**Abstract:**

This document specifies the means by which SCA composites and components, as defined in the SCA Assembly Specification [**SCA-Assembly**], connect to and access services using a messaging protocol. The connectivity is based on the Java Messaging Service [**JMS**] and is provided by a binding.jms element which applies to the references and services of an SCA component or composite.

The JMS binding provides JMS-specific details of the connection to the required JMS resources. It supports the use of Queue and Topic type destinations.

The binding is especially well suited for use by services and references of composites that are directly deployed, as opposed to composites that are used as implementations of higher-level components. Services and references of deployed composites become system-level services and references, which are intended to be used by non-SCA clients.

**Status:**

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

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "[Send A Comment](#)" button on the Technical Committee's web page at <http://www.oasis-open.org/committees/sca-bindings/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<http://www.oasis-open.org/committees/sca-bindings/ipr.php>).

**Citation Format:**

When referencing this specification the following citation format should be used:

**[SCA-JMSBINDING-v1.1—OASIS Committee Specification Draft 05, JMSBinding]**

*Service Component Architecture JMS Binding Specification Version 1.1, 14 July 2011. OASIS Committee Specification Draft 06 / Public Review Draft 04. <http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-spec-csprd04.html>, November 2010.*

---

## Notices

Copyright © OASIS~~© 2006, 2010.~~ [Open 2011](#). All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full ~~Policy~~ [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", "SCA" and "Service Component Architecture" are trademarks of ~~OASIS~~ [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

---

# Table of Contents

1	Introduction.....	5
1.1	Terminology.....	5
1.2	Normative References.....	5
1.3	Non-Normative References.....	6
1.4	Naming Conventions.....	6
1.5	Testcases.....	6
2	Messaging Bindings.....	7
3	JMS Binding Schema.....	8
3.1	Extensibility.....	13
3.2	JMS Message Headers and User Properties.....	13
3.3	JMS Message Selection.....	13
4	Operation Selectors and Wire Formats.....	14
4.1	Default Operation Selection.....	14
4.2	Default Wire Format.....	15
4.2.1	Example of default wire format.....	15
5	Policy.....	17
6	Message Exchange Patterns.....	18
6.1	One-way message exchange (no Callbacks).....	18
6.2	Request/response message exchange (no Callbacks).....	18
6.3	JMS User Properties.....	19
6.4	Callbacks.....	19
6.4.1	Invocation of operations on a bidirectional interface.....	19
6.4.2	Invocation of operations on a callback interface.....	20
6.4.3	Use of JMSReplyTo for callbacks for non-SCA JMS applications.....	20
7	Examples.....	21
7.1	Minimal Binding Example.....	21
7.2	URI Binding Example.....	21
7.3	Binding with Existing Resources Example.....	21
7.4	Resource Creation Example.....	22
7.5	Request/Response Example.....	22
7.6	Subscription with Selector Example.....	23
7.7	Policy Set Example.....	23
8	Conformance.....	25
8.1	SCA JMS Binding XML Document.....	25
8.2	SCA Runtime.....	25
A.	JMS XML Binding Schema: sca-binding-jms-1.1.xsd.....	26
B.	Conformance Items.....	30
C.	Acknowledgements.....	35
D.	Revision History.....	36

# 1 Introduction

This document specifies the means by which SCA composites and components, as defined in the SCA Assembly Specification [**SCA-Assembly**], connect to and access services using a messaging protocol. The connectivity is based on the Java Messaging Service [**JMS**] and is provided by a binding.jms element which applies to the references and services of an SCA component or composite.

The JMS binding provides JMS-specific details of the connection to the required JMS resources. It supports the use of Queue and Topic type destinations.

The binding is especially well suited for use by services and references of composites that are directly deployed, as opposed to composites that are used as implementations of higher-level components. Services and references of deployed composites become system-level services and references, which are intended to be used by non-SCA clients.

## 1.1 Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC Keywords \[RFC2119\]](#).

This specification uses predefined namespace prefixes throughout; they are given in the following list. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

Prefix	Namespace	Notes
xs	"http://www.w3.org/2001/XMLSchema"	Defined by XML Schema 1.0 specification
sca	"http://docs.oasis-open.org/ns/opencsa/sca/200912"	Defined by the SCA specifications

Table 1-1: Prefixes and Namespaces used in this specification

## 1.2 Normative References

- [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- [JMS] Java™ Message Service Specification v1.1  
<http://www.oracle.com/technetwork/java/jms/index.html>
- [JNDI] [Java™ Naming and Directory Interface](http://www.oracle.com/technetwork/java/jndi/index.html)  
<http://www.oracle.com/technetwork/java/jndi/index.html>
- [WSDL] E. Christensen et al, *Web Service Description Language (WSDL) 1.1*, <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>, W3C Note, March 15 2001.  
R. Chinnici et al, *Web Service Description Language (WSDL) Version 2.0 Part 1: Core Language*, <http://www.w3.org/TR/2007/REC-wsdl20-20070626/>, W3C Recommendation, June 26 2007.

32	<b>[JCA15]</b>	J2EE Connector Architecture Specification Version 1.5 <a href="http://java.sun.com/j2ee/connector/">http://java.sun.com/j2ee/connector/</a>
33		
34	<b>[IETFJMS]</b>	M. Phillips, P. <del>Easton</del> Adams, D. Rokicki, E. Johnson, <i>URI Scheme for Java™ Message Service 1.0</i> , <a href="http://www.ietf.org/rfc/rfc6167.txt">http://www.ietf.org/rfc/rfc6167.txt</a> , IETF <del>Internet-Draft</del> September 2010 <sup>†</sup> <a href="http://www.ietf.org/rfc/rfc6167.txt">RFC 6167</a> , April 2011.
35		
36		
37	<b>[SCA-Assembly]</b>	OASIS Committee <a href="http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-csd07.pdf">Specification</a> Draft <del>0507</del> , <i>Service Component Architecture Assembly Model Specification Version 1.1</i> , January 2010 <sup>1</sup>
38		
39		<a href="http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-csd07.pdf">http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-csd07.pdf</a>
40		
41	<b>[SCA-Policy]</b>	OASIS Committee Draft 04, <i>SCA Policy Framework Specification Version 1.1</i> , September 2010
42		
43		<a href="http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd04.pdf">http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd04.pdf</a>

### 44 1.3 Non-Normative References

45 [N/A](#)

46 [\[SCA-JMSTest\]](#) [OASIS Committee Specification Draft 01, SCA JMS Binding v1.1 TestCases](http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-testcases-1.0-csd01.pdf)  
47 [Version 1.0, November 2010,](http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-testcases-1.0-csd01.pdf)  
48 [http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-testcases-](http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-testcases-1.0-csd01.pdf)  
49 [1.0-csd01.pdf](http://docs.oasis-open.org/opencsa/sca-bindings/sca-jmsbinding-1.1-testcases-1.0-csd01.pdf)

### 50 1.4 Naming Conventions

51 The naming conventions used by artefacts defined in this specification are:

- 52 • The naming conventions defined by section 1.3 of the [SCA Assembly Specification \[SCA-Assembly\]](#).
- 53 • Where the names of elements and attributes consist partially or wholly of acronyms, the letters of the  
54 acronyms use the same case. When the acronym appears at the start of the name of an element or  
55 an attribute, or after a period, it is in lower case. If it appears elsewhere in the name of an element or  
56 an attribute, it is in upper case. For example, an attribute might be named "uri" or "jndiURL".
- 57 • Where the names of types consist partially or wholly of acronyms, the letters of the acronyms are in  
58 all upper case. For example, an XML Schema type might be named "JCABinding" or "MessageID".
- 59 • Values, including local parts of QName values, follow the rules for names of elements and attributes  
60 as stated above, with the exception that the letters of acronyms are in all upper case. For example, a  
61 value might be "JMSDefault" or "namespaceURI".

### 62 1.5 Testcases

63 [SCA JMS Binding TestCases Version 1.1 \[SCA-JMSTest\]](#) defines test cases for this specification. The  
64 [TestCases represent a series of tests that SCA runtimes are expected to pass in order to claim](#)  
65 [conformance to the requirements of this specification.](#)

---

<sup>†</sup> ~~Note that this URI scheme is currently in draft. The reference for this specification will be updated when the IETF standard is finalized~~

---

## 66 **2 Messaging Bindings**

67 Messaging bindings form a category of SCA bindings that represent the interaction of SCA composites  
68 with messaging providers. It is felt that documenting, and following this pattern is beneficial for  
69 implementers of messaging bindings, although it is not strictly necessary.

70 This pattern is embodied in the JMS binding, described later.

71 Messaging bindings utilize operation selector and wire format elements to provide the mapping from the  
72 native messaging format to an invocation on the target component. A default operation selection and  
73 data binding behavior is specified.

74 In addition, each operation in the interface associated with the service or reference can have properties  
75 specified, that influence the way native messages are processed depending on the operation being  
76 invoked.

77

## 3 JMS Binding Schema

78

The JMS binding element is defined by the pseudo-schema in Snippet 3-1.

```

79 <binding.jms correlationScheme="QName"?
80     initialContextFactory="xs:anyURI"?
81     jndiURL="xs:anyURI"?
82     name="NCName"?
83     requires="list of QName"?
84     policySets="list of QName"?
85     uri="xs:anyURI"? >
86 <destination jndiName="xs:anyURI"? type="queue or topic"?
87     create="always or never or ifNotExist"?>
88     <property name="NMTOKEN" type="string"?>*
89 </destination?>
90 <connectionFactory jndiName="xs:anyURI"?
91     create="always or never or ifNotExist"?>
92     <property name="NMTOKEN" type="string"?>*
93 </connectionFactory?>
94 <activationSpec jndiName="xs:anyURI"?
95     create="always or never or ifNotExist"?>
96     <property name="NMTOKEN" type="string"?>*
97 </activationSpec?>
98
99 <response>
100     <destination jndiName="xs:anyURI"? type="queue or topic"?
101         create="always or never or ifNotExist"?>
102         <property name="NMTOKEN" type="string"?>*
103     </destination?>
104     <connectionFactory jndiName="xs:anyURI"?
105         create="always or never or ifNotExist"?>
106         <property name="NMTOKEN" type="string"?>*
107     </connectionFactory?>
108     <activationSpec jndiName="xs:anyURI"?
109         create="always or never or ifNotExist"?>
110         <property name="NMTOKEN" type="string"?>*
111     </activationSpec?>
112     <wireFormat/>?
113 </response?>
114
115 <resourceAdapter name="NMTOKEN">?
116     <property name="NMTOKEN" type="string"?>*
117 </resourceAdapter?>
118
119 <headers type="string"?
120     deliveryMode="persistent or nonpersistent"?
121     timeToLive="long"?
122     priority="0 .. 9"?>
123     <property name="NMTOKEN" type="boolean or byte or .. or String"?>*
124 </headers?>
125
126 <messageSelection selector="string"?>
127     <property name="NMTOKEN" type="string"?>*
128 </messageSelection?>
129
130 <operationProperties name="string" selectedOperation="string"?>
131     <property name="NMTOKEN" type="string"?>*
132     <headers type="string"?
133         deliveryMode="persistent or nonpersistent"?
134         timeToLive="long"?
135         priority="0 .. 9"?>

```



```

136         <property name="NMOKEN" type="boolean or byte or .. or String"?>*
137     </headers>?
138 </operationProperties>*
139
140     <wireFormat ... />?
141     <operationSelector ... />?
142 </binding.jms>

```

143 *Snippet 3-1: binding.jms Pseudo-Schema*

144 The binding can be used in one of two ways, either identifying existing [JMS \[JMS\]](#) resources using JNDI  
145 [\[JNDI\]](#) names, or providing the required information to enable the JMS resources to be created.

146 The `binding.jms` element has the attributes:

- 147 • **`/binding.jms`** – This is the JMS binding element. The element is extensible so that JMS binding  
148 implementers can add additional JMS provider-specific attributes and elements although such  
149 extensions are not guaranteed to be portable across runtimes.
- 150 • **`/binding.jms/@uri`** – as defined in the [SCA Assembly Specification \[SCA-Assembly\]](#). This attribute  
151 identifies the destination, connection factory or activation spec, and other properties to be used to  
152 send/receive the JMS message. There is an implicit `@create="never"` for the resources referred to  
153 in the `@uri` attribute. Message header properties and the message selector set via the `@uri` attribute  
154 take precedence over those specified in binding elements as defined in section 3.2.

155 **The value of the `@uri` attribute MUST have the format defined by the IETF URI Scheme for Java™  
156 Message Service 1.0 [IETFJMS] [BJM30001].**

157 Snippet 3-2 illustrates the structure of the URI and the set of property names that have specific  
158 semantics:

```

159 jms:jndi:<jms-dest>?
160 jndiURL=<jndi-url> &
161 jndiInitialContextFactory=<jndi-initial-context-factory> &
162 jndiConnectionFactoryName=<Connection-Factory-Name> &
163 deliveryMode=<Delivery-Mode> &
164 timeToLive=<Time-To-Live> &
165 priority=<Priority> &
166 selector=<Message-Selector> &
167 <param-name>=<param-value> & ...

```

168 *Snippet 3-2: JMS URI Structure*

169 **When the `@uri` attribute is specified, the SCA runtime MUST raise an error if the referenced  
170 resources do not already exist [BJM30002].**

171 **When the `@uri` attribute is specified, the `destination` element MUST NOT be present  
172 [BJM30034].**

- 173 • **`/binding.jms/@name`** - as defined in the [SCA Assembly Specification \[SCA-Assembly\]](#).
- 174 • **`/binding.jms/@requires`** - as defined in the [SCA Assembly Specification \[SCA-Assembly\]](#).
- 175 • **`/binding.jms/@policySets`** - as defined in the [SCA Assembly Specification \[SCA-Assembly\]](#).
- 176 • **`/binding.jms/@correlationScheme`** – identifies the correlation scheme used when sending reply or  
177 callback messages, default value is `"sca:messageID"`. Three specific behaviours are provided.  
178 `"sca:messageID"` indicates that response messages can be correlated with their requests by  
179 looking for the request's `messageID` header value in the response's `correlationID` header;  
180 `"sca:correlationID"` indicates that response messages can be correlated with their requests by  
181 looking for the request's `correlationID` header value in the response's `correlationID` header;  
182 `"sca:none"` indicates that the response's `correlationID` header is not to be used for this purpose and  
183 some other means is used for the correlation.

184 **If the value of the `@correlationScheme` attribute is `"sca:messageID"` the SCA runtime MUST set  
185 the correlation ID of replies to the message ID of the corresponding request [BJM30003].**

186 If the value of the @correlationScheme attribute is "sca:correlationID" the SCA runtime  
187 MUST set the correlation ID of replies to the correlation ID of the corresponding request [BJM30004].

188 If the value of the @correlationScheme attribute is "sca:correlationID" the SCA runtime  
189 MUST set a non-null correlation ID value in requests that it sends [BJM30007].

190 If the value of the @correlationScheme attribute is "sca:none" the SCA runtime MUST NOT set  
191 the correlation ID in responses that it sends [BJM30005].

192 ~~[BJM30006].~~

193 [SCA runtimes supporting other correlation schemes can allow additional values for the](#)  
194 [@correlationScheme attribute.](#)

- 195 • **/binding.jms/@initialContextFactory** – the name of the JNDI initial context factory.
- 196 • **/binding.jms/@jndiURL** – the URL for the JNDI provider.
- 197 • **/binding.jms/destination** – identifies the destination that is to be used to process requests by this  
198 binding.
- 199 • **/binding.jms/destination/@type** - the type of the request destination. Valid values are "queue" and  
200 "topic". The default value is "queue".

201 Whatever the value of the destination/@type attribute, the SCA runtime MUST ensure a single  
202 response is delivered for request/response operations [BJM30010].

- 203 • **binding.jms/destination/@jndiName** – the JNDI name of the JMS Destination that the binding uses  
204 to send or receive messages. The behaviour of this attribute is determined by the value of the  
205 @create attribute as follows:

- 206 – If the @create attribute value for a destination, connectionFactory or activationSpec  
207 element is "always" and the @jndiName attribute is present and the resource cannot be created  
208 at the location specified by the @jndiName attribute then the SCA runtime MUST raise an error  
209 [BJM30011].

- 210 If the @create attribute value for a destination, connectionFactory or activationSpec  
211 element is "always" and the @jndiName attribute is not present and the resource cannot be  
212 created, then the SCA runtime MUST raise an error [BJM30037].

213 If the @jndiName attribute is omitted this specification places no restriction on the JNDI location  
214 of the created resource.

- 215 – If the @create attribute value for a destination, connectionFactory or activationSpec  
216 element is "ifNotExist" then the @jndiName attribute MUST specify the location of the  
217 possibly existing resource [BJM30012].

- 218 If the @create attribute value for a destination, connectionFactory or activationSpec  
219 element is "ifNotExist" and the resource does not exist at the location identified by the  
220 @jndiName attribute and cannot be created there then the SCA runtime MUST raise an error  
221 [BJM30013].

- 222 If the @create attribute value for a destination, connectionFactory or activationSpec  
223 element is "ifNotExist" and the @jndiName attribute refers to an existing resource that is not  
224 a JMS Destination of the appropriate type, a JMS connection factory or a JMS activation spec  
225 respectively then the SCA runtime MUST raise an error [BJM30014].

- 226 – If the @create attribute value for a destination, connectionFactory or activationSpec  
227 element is "never" and the @jndiName attribute is not specified, or the resource is not present  
228 at the location identified by the @jndiName attribute, or the location refers to a resource of an  
229 incorrect type then the SCA runtime MUST raise an error [BJM30015].

- 230 • **/binding.jms/destination/@create** – indicates whether the destination should be created when the  
231 containing composite is deployed. Valid values are "always", "never" and "ifNotExist".

232 "always" indicates that new resources are created for use by this binding; "never" indicates that  
233 existing resources are used and none created; "ifNotExist" indicates that if the resources  
234 already exist those are used, otherwise new ones are created. Refer to the  
235 destination/@jndiName attribute for a detailed definition of each case. The default value is  
236 "ifNotExist".

- 237 • **/binding.jms/destination/property** – defines properties to be used to create the destination, if  
238 required.
- 239 • **/binding.jms/connectionFactory** – identifies the connection factory that the binding uses to process  
240 request messages. The attributes of this element follow the rules defined for the destination  
241 element.

242 A binding.jms element MUST NOT include both a connectionFactory element and an  
243 activationSpec element [BJM30017].

244 When the connectionFactory element is present as a child of the binding.jms element, then  
245 the destination MUST be defined either by the destination element child of the binding.jms  
246 element or the @uri attribute of the binding.jms element [BJM30018].

- 247 • **/binding.jms/activationSpec** – identifies the activation spec that the binding uses to connect to a  
248 JMS destination to process request messages. The attributes of this element follow the rules defined  
249 for the destination element.

250 If the activationSpec element is present as a child of the binding.jms element and the  
251 destination is also specified via a destination element child of the binding.jms element or the  
252 @uri attribute of the binding.jms element then it MUST refer to the same JMS destination as the  
253 activationSpec [BJM30019].

254 The activationSpec element MUST NOT be present when the binding is being used for an SCA  
255 reference [BJM30020].

- 256 • **/binding.jms/response** – defines the resources used for handling response messages (receiving  
257 responses for a reference, and sending responses from a service).
- 258 • **/binding.jms/response/destination** – identifies the destination that is to be used to process  
259 responses by this binding. Attributes follow the rules defined for the parent's destination element.  
260 For a service, this destination is used to send responses to messages that have a null value for the  
261 JMSReplyTo destination. For a reference, this destination is used to receive reply messages
- 262 • **/binding.jms/response/connectionFactory** – identifies the connection factory that the binding uses  
263 to process response messages. The attributes of this element follow those defined for the  
264 destination element.

265 A response element MUST NOT include both a connectionFactory element and an  
266 activationSpec element [BJM30021].

- 267 • **/binding.jms/response/activationSpec** – identifies the activation spec that the binding uses to  
268 connect to a JMS destination to process response messages. The attributes of this element follow  
269 those defined for the destination element.

270 If a response/destination and response/activationSpec element are both specified they  
271 MUST refer to the same JMS destination [BJM30022].

272 The response/activationSpec element MUST NOT be present when the binding is being used  
273 for an SCA service [BJM30023].

- 274 • **/binding.jms/response/wireFormat** – identifies the wire format used by responses sent or received  
275 by this binding. This value overrides the wireFormat specified at the binding level. Wire formats for  
276 this binding are described in Section 4.
- 277 • **/binding.jms/headers** – this element specifies values to be set for standard JMS headers. These  
278 values apply to requests from a reference and responses from a service. Section 3.2 defines the  
279 priority rules for determining the values for JMS headers and user properties.

- 280 • ***/binding.jms/headers/@type, @deliveryMode, @timeToLive, @priority*** – specifies the value to  
 281 use for the JMS header property JMSType, JMSDeliveryMode, JMSTimeToLive or JMSPriority  
 282 respectively. Valid values for @deliveryMode are "persistent" and "nonpersistent",  
 283 corresponding to the values defined in the JMS Specification [JMS] for the JMSDeliveryMode  
 284 message header, with "persistent" being the default; valid values for @priority are "0" to  
 285 "9", where "0" indicates lowest priority and "9" highest priority, with "4" being the default; valid  
 286 values for @timeToLive are positive integers, with 0 indicating unlimited time and being the default  
 287 value.
- 288 • ***/binding.jms/headers/property*** – specifies the value and type for the named JMS user property.
- 289 • ***/binding.jms/messageSelection*** - this element specifies JMS message selection options. This  
 290 element applies to a service receiving messages from the request destination or for a reference  
 291 receiving messages from the callback or reply-to destination.
- 292 • ***/binding.jms/messageSelection/@selector*** - specifies the value to use for the JMS message  
 293 selector. Section 3.3 defines the priority rules for determining the values for the message selector.
- 294 • ***/binding.jms/resourceAdapter*** – specifies name, type and properties of the Resource Adapter Java  
 295 bean. [The resource adapter and SCA runtime together define the set of valid properties for](#)  
 296 [configuring the resource adapter via the JMS binding.](#)
- 297 The `resourceAdapter` element MUST be present when JMS resources are to be created for a JMS  
 298 provider that implements the JCA 1.5 Specification [JCA15] specification, and is ignored otherwise  
 299 [BJM30031].
- 300 ~~[BJM30028].~~
- 301 For JMS providers that do not implement the [JCA 1.5 specification \[JCA15\]](#), information necessary for  
 302 resource creation can be added in provider-specific elements or attributes allowed by the extensibility  
 303 of the `binding.jms` element.
- 304 • ***/binding.jms/operationProperties*** – specifies various properties that are specific to the processing  
 305 of a particular operation.
- 306 • ***/binding.jms/operationProperties/@name*** – The name of the operation in the interface.
- 307 • ***/binding.jms/operationProperties/@selectedOperation*** – The value generated by the  
 308 `operationSelector` that corresponds to the operation in the service or reference interface  
 309 identified by the `operationProperties/@name` attribute. If this attribute is omitted then the value  
 310 defaults to the value of the `operationProperties/@name` attribute.
- 311 The value of the ***operationProperties/@selectedOperation*** attribute MUST be unique across the  
 312 containing `binding.jms` element [BJM30029].
- 313 • ***/binding.jms/operationProperties/property*** – specifies properties specific to this operation. These  
 314 properties are intended to be used to parameterize the `wireFormat` identified for the binding for a  
 315 particular operation.
- 316 ~~[BJM30030].~~
- 317 • ***/binding.jms/operationProperties/headers*** – this element specifies values to be set for standard  
 318 JMS headers. These values apply to requests from a reference and responses from a service.  
 319 Section 3.2 defines the priority rules for determining the values for JMS headers and user properties.
- 320 • ***/binding.jms/operationProperties/headers/@type, @deliveryMode, @timeToLive, @priority*** –  
 321 specifies the value to use for the JMS header property JMSType, JMSDeliveryMode, JMSTimeToLive  
 322 or JMSPriority, respectively. Refer to the description of the `binding.jms/headers` element for the  
 323 valid values for these attributes.
- 324 • ***/binding.jms/operationProperties/headers/property*** – specifies the value and type for the named  
 325 JMS user property.
- 326 • ***/binding.jms/wireFormat*** – identifies the wire format used by requests and responses sent or  
 327 received by this binding. Wire formats for this binding are described in Section 4.

- 328 • ***/binding.jms/operationSelector*** – identifies the operation selector used when receiving requests for  
329 a service. If specified for a reference this provides the default operation selector for callbacks if not  
330 specified via a callback service element. Operation selectors for this binding are described in Section  
331 3.2.

332 The `binding.jms` element MUST conform to the XML schema defined in `sca-binding-jms-1.1.xsd`  
333 [BJM30036].

### 334 3.1 Extensibility

335 The JMS binding allows further customization of the binding element and its subelements with vendor  
336 specific attributes or elements. This is done by providing extension points in the schema; refer to  
337 Appendix A, “JMS XML Binding Schema: `sca-binding-jms-1.1.xsd`” for the locations of these extension  
338 points.

### 339 3.2 JMS Message Headers and User Properties

340 The JMS binding can be configured to specify that JMS headers are set to specific values in messages  
341 sent by the SCA runtime. The binding provides several places where JMS message headers and user  
342 properties can be specified at different levels of granularity.

343 The type of the JMS user property is specified via the `property/@type` attribute using one of the values  
344 define in the JMS specification [**JMS**]: "boolean", "byte", "short", "int", "long", "float", "double", "String" (the  
345 default), or "xs:string". "xs:string" and "String" both represent the String user property type, "xs:string" is  
346 for backward compatibility only and its use is deprecated.

347 When sending messages for a JMS binding, the SCA runtime MUST set each of the `JMSType`,  
348 `JMSDeliveryMode`, `JMSTimeToLive` and `JMSPriority` headers to values specified in the binding definition  
349 in the following priority order:

- 350 1) the value for the header specified in the `@uri` attribute (highest priority);
- 351 2) the value for the header specified in the `operationProperties/headers` element matching the  
352 operation being invoked;
- 353 3) the value for the header specified in the `headers` element;
- 354 4) the default value for the header as specified by the definition of the `binding.jms/headers`  
355 element (lowest priority) [BJM30024].

356 When sending messages for a JMS binding, the SCA runtime MUST set each named user property with  
357 type and value specified in the binding definition in the following priority order:

- 358 1) the type and value for the named user property specified in an  
359 `operationProperties/headers/property` element matching the name of the operation being  
360 invoked (highest priority);
- 361 2) the type and value for the named user property specified in a `headers/property` element (lowest  
362 priority) [BJM30025].

### 363 3.3 JMS Message Selection

364 Message selectors can be specified for the JMS binding to receive a specific subset of messages from a  
365 given destination, such that only messages that match the selector are delivered to a given JMS binding.  
366 This allows more than one JMS binding to share a destination.

367 When receiving messages for a JMS binding, the SCA runtime MUST use a message selector if specified  
368 in the binding definition in the following priority order:

- 369 1) the value for the message selector specified in the `@uri` attribute value's "selector" parameter  
370 (highest priority);
- 371 2) the value for the message selector specified in the `messageSelection/@selector` attribute;
- 372 3) otherwise no message selector is used (lowest priority) [BJM30026].

373

## 4 Operation Selectors and Wire Formats

374 In general messaging providers deal with message formats and destinations. There is not usually a built-  
375 in concept of “operation” that corresponds to that defined in a [WSDL \[WSDL\]](#) portType. Messages have  
376 a wire format which corresponds in some way to the schema of an input or output message of an  
377 operation in the interface of a service or reference, however additional information is required in order for  
378 an SCA runtime to know how to identify the operation and understand the wire format of messages.

379 The process of identifying the operation to be invoked is *operation selection*; the information that  
380 describes the contents of messages is a *wire format*. The `binding` element as described in the [SCA  
381 Assembly Specification \[SCA-Assembly\]](#) provides the means to identify specific operation selection via  
382 the `operationSelector` element and the wire format of messages received and to be sent using the  
383 `wireFormat` element. [The `operationSelector` and `wireFormat` elements allow a binding element  
384 to specify behaviour defined by the binding specification or custom behaviour provided by an SCA  
385 runtime.](#)

386 When the service with a JMS binding receives a message, the SCA runtime resolves the name of the  
387 operation in the service's interface that is to be invoked by using the `operationSelector` and  
388 `operationProperties` elements defined for the binding. The *resolved operation name* is defined as  
389 follows:

- 390 • If the selected operation name generated by the `operationSelector` matches the value of an  
391 `operationProperties/@selectedOperation` attribute then the resolved operation name is the  
392 value of the `operationProperties/@name` attribute.
- 393 • Otherwise the resolved operation name is the selected operation name generated by the  
394 `operationSelector`.

395 When a message is received at an SCA service with JMS binding and the resolved operation name is in  
396 the target component's interface, the SCA runtime MUST invoke the target component using the resolved  
397 operation name [BJM40010].

398 When a message is received at an SCA service with JMS binding and the resolved operation name is not  
399 in the target component's interface the SCA runtime MUST raise an error [BJM40011].

400 No standard means is provided for linking the `wireFormat` or `operationSelector` elements with the  
401 runtime components that implement their behavior.

402 The following sections describe the default `operationSelector` and `wireFormat` for a JMS binding.  
403 ~~[BJM40004].~~

### 4.1 Default Operation Selection

405 The following defines the **default operation selection algorithm** when receiving a request at a service,  
406 or a callback at a reference. When using the default operation selection algorithm, the selected operation  
407 name is determined as follows:

- 408 • If there is only one operation on the service's interface, then that operation is the selected operation  
409 name;
- 410 • Otherwise, if the JMS user property “`scaOperationName`” is present, then the value of that user  
411 property is used as the selected operation name;
- 412 • Otherwise, if the message is a JMS text or bytes message containing XML, then the selected  
413 operation name is the local name of the root element of the XML payload;
- 414 • Otherwise, the selected operation name is “`onMessage`”.

415 When a `binding.jms` element specifies the `operationSelector.jmsDefault` element, the SCA  
416 runtime MUST use the default operation selection algorithm to determine the selected operation  
417 [BJM40008].

418 If no `operationSelector` element is specified then SCA runtimes MUST use  
419 `operationSelector.jmsDefault` as the default [BJM40002].

## 420 4.2 Default Wire Format

421 The default wire format maps between a `JMSMessage` and the object(s) expected by the component  
422 implementation. We encourage component implementers to avoid exposure of [JMS \[JMS\]](#) APIs to  
423 component implementations, however in the case of an existing implementation that expects a  
424 `JMSMessage`, this provides for simple reuse of that as an SCA component.

425 When using the default wire format, the message body is mapped to the parameters or return value of the  
426 target operation as follows:

- 427 • If there is a single parameter that is a `JMSMessage`, then the `JMSMessage` is passed as is.
- 428 • Otherwise, if the `JMSMessage` is not a JMS text message or bytes message containing XML it is  
429 invalid.
- 430 • Otherwise if there is a single parameter, or for the return value, the JMS text or bytes XML payload is  
431 the XML serialization of that parameter according to the WSDL schema for the message.
- 432 • Otherwise the multiple parameters are encoded in XML using the document wrapped style, according  
433 to the WSDL schema for the message.

434 When a `binding.jms` element specifies the `wireFormat.jmsDefault` element, the SCA runtime  
435 MUST use the default wire format [BJM40009].

436 When using the default wire format to send request messages, if there is a single parameter and the  
437 interface includes more than one operation, the SCA runtime MUST set the JMS user property  
438 `"scaOperationName"` to the name of the operation being invoked [BJM40003].

439 When using the default wire format an SCA runtime MUST be able to receive both JMS text and bytes  
440 messages [BJM40005].

441 When using the default wire format an SCA runtime MUST send either a JMS text or a JMS bytes  
442 message [BJM40006].

443 ~~[BJM40007].~~

444 If no `wireFormat` element is specified in a JMS binding then SCA runtimes MUST use  
445 `wireFormat.jmsDefault` as the default [BJM40004].

446 [The default wire format allows a choice of text or bytes format when sending messages; an SCA runtime  
447 can restrict this to one or other via additional configuration.](#)

### 448 4.2.1 Example of default wire format

449 For the interface definition in Snippet 4-1:

```
450 <wsdl:definitions name="Coordinates"  
451 targetNamespace="http://tempuri.org/coordinates"  
452 xmlns:tns="http://tempuri.org/coordinates"  
453 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"  
454 xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
455   <wsdl:types>  
456     <xsd:schema targetNamespace="http://tempuri.org/coordinates">  
457       <xsd:element name="setCoordinates">  
458         <xsd:complexType>  
459           <xsd:sequence>  
460             <xsd:element name="x" type="xsd:int"/>  
461             <xsd:element name="y" type="xsd:int"/>  
462           </xsd:sequence>  
463         </xsd:complexType>  
464       </xsd:element>  
465     </xsd:schema>  
466   </wsdl:types>  
467
```

```
468 <wsdl:message name="setCoordinatesRequestMsg">
469   <wsdl:part element="tns:setCoordinates" name="setCoordinatesParameters"/>
470 </wsdl:message>
471
472 <wsdl:portType name="Coordinates">
473   <wsdl:operation name="setCoordinates">
474     <wsdl:input message="tns:setCoordinatesRequestMsg"
475       name="setCoordinatesRequest"/>
476   </wsdl:operation>
477 </wsdl:portType>
478 </wsdl:definitions>
```

479 *Snippet 4-1: Example WSDL Interface Definition*

480 When the `setCoordinates` operation is invoked via a reference with a JMS binding that uses the  
481 default wire format, the message sent from the JMS binding is a JMS text or bytes message with the  
482 content shown in Snippet 4-2:

```
483 <setCoordinates xmlns="http://tempuri.org/coordinates">
484   <x>10</x>
485   <y>5</y>
486 </setCoordinates>
```

487 *Snippet 4-2: JMS Message Content for setCoordinates Operation of Snippet 4-1*



---

## 488 5 Policy

489 The JMS binding provides attributes that control the sending of messages, requests from references and  
490 replies from services. These values can be set directly on the binding element for a particular service or  
491 reference, or they can be set using policy intents. An example of setting these via intents is shown later.

492 **JMS binding implementations MUST support the JMS intent [BJM50001].**

493 **The JMS intent MUST always be included in the @alwaysProvides attribute of the JMS bindingType**  
494 **[BJM50002]**

495 The following standard intents can also be supported by JMS binding implementations, by inclusion in the  
496 @alwaysProvides or @mayProvides attribute of the JMS bindingType:

- 497 • atLeastOnce
- 498 • atMostOnce
- 499 • ordered

500 The atLeastOnce, atMostOnce and ordered intents are defined in the [SCA Policy Specification](#)  
501 [\[SCA-Policy\]](#) document in section 8, "Reliability Policy".

502 This specification does not define a fixed relationship between the reliability intents and the persistence of  
503 JMS messages. Deployers/assemblers can configure a nonpersistent delivery mode via the  
504 @deliveryMode or @uri attribute, in order to provide higher performance with a decreased quality of  
505 service. However a binding.jms element configured with a nonpersistent delivery mode might not be able  
506 to satisfy the atLeastOnce policy intent. The [SCA Policy Specification \[SCA-Policy\]](#) requires that an  
507 error be raised if the SCA runtime is unable to support the intents on a binding in combination with the  
508 specific configuration of that binding.

---

## 509 6 Message Exchange Patterns

510 This section describes the message exchange patterns that are possible when using the JMS binding,  
511 including one-way, request/response and callbacks. [JMS \[JMS\]](#) has a looser concept of message  
512 exchange patterns than WSDL, so this section explains how JMS messages that are sent and received  
513 by the SCA runtime relate to the WSDL input/output messages. Each operation in a WSDL interface is  
514 either one-way or request/response. Callback interfaces can include both one-way and request/response  
515 operations.

### 516 6.1 One-way message exchange (no Callbacks)

517 A one-way message exchange is one where a request message is sent that does not require or expect a  
518 corresponding response message. These are represented in WSDL as an operation with an `input`  
519 element and no `output` elements and no `fault` elements. [The JMS specification provides the](#)  
520 [JMSReplyTo header as the way for a JMS application to identify the destination on which replies or other](#)  
521 [messages are to be placed that relate to the one being sent. For one-way requests sent by SCA](#)  
522 [references with unidirectional interfaces, the JMSReplyTo will not usually be set as no reply or other](#)  
523 [related message is expected.](#)

524 [\[BJM60004\].](#)

525 For an SCA service with a JMS binding and unidirectional interface, when a request message is received  
526 as part of a one-way MEP, the SCA runtime MUST ignore the `JMSReplyTo` destination header in the  
527 JMS message, and not raise an error [\[BJM60002\].](#)

528 The use of one-way exchanges when using a bidirectional interface is described in section 6.4.

### 529 6.2 Request/response message exchange (no Callbacks)

530 A request/response message exchange is one where a request message is sent and a response  
531 message is expected, possibly identified by its correlation identifier. These are represented in WSDL as  
532 an operation with an `input` element and an `output` and/or a `fault` element.

533 For an SCA reference with a JMS binding, when a request message is sent as part of a request/response  
534 MEP, and the JMS binding has a `response` element with a `destination` defined, then the SCA  
535 runtime MUST use that destination for the `JMSReplyTo` header in the JMS message it creates for the  
536 request [\[BJM60004\].](#)

537 For an SCA reference with a JMS binding, when a request message is sent as part of a request/response  
538 MEP, and the JMS binding does not have a `response` element with a `destination` defined, the SCA  
539 runtime MUST provide an appropriate destination on which to receive response messages and use that  
540 destination for the `JMSReplyTo` header in the JMS message it creates for the request [\[BJM60005\].](#)

541 For an SCA reference with a JMS binding that does not have a destination specified via the `response`  
542 element, the SCA runtime MUST either receive response messages as defined by the binding's  
543 `@correlationScheme` attribute, or use a unique destination for each request/response interaction  
544 [\[BJM60006\].](#)

545 For an SCA reference with a JMS binding that has a destination specified via the `response` element, the  
546 SCA runtime MUST receive response messages as defined by the binding's `@correlationScheme`  
547 attribute [\[BJM60003\].](#)

548 For an SCA service with a JMS binding, when a response message is sent as part of a request/response  
549 MEP where the request message included a non-null `JMSReplyTo` destination, the SCA runtime MUST  
550 send the response message to that destination [\[BJM60007\].](#)

551 For an SCA service with a JMS binding, when a response message is sent as part of a request/response  
552 MEP where the request message included a null `JMSReplyTo` destination and the JMS binding includes

553 a `response/destination` element the SCA runtime MUST send the response message to that  
554 destination [BJM60008].

555 For an SCA service with a JMS binding, when a `request` message is `received` as part of a  
556 request/response MEP where the request message includes a null `JMSReplyTo` destination and the JMS  
557 binding does not include a `response/destination` then the SCA runtime **MUST NOT process the request**  
558 **and MUST raise an error** [BJM60009].

559 For an SCA service with a JMS binding, when a response message is sent as part of a request/response  
560 MEP the SCA runtime MUST set the correlation identifier in the JMS message that it creates for the  
561 response as defined by the JMS binding's `@correlationScheme` attribute [BJM60010].

562 The use of request/response exchanges when using a bidirectional interface is described in section 6.4.

## 563 6.3 JMS User Properties

564 This protocol assigns specific behavior to JMS user properties:

- 565 • "`scaCallbackDestination`" holds a JMS URI that identifies the Destination to which callback  
566 messages are sent, in the format defined by the IETF URI Scheme for Java™ Message Service 1.0  
567 [IETFJMS].

## 568 6.4 Callbacks

569 Callbacks are SCA's way of representing bidirectional interfaces, where messages are sent in both  
570 directions between a client and a service. A callback is the invocation of an operation on a service's  
571 callback interface. A callback operation can be one-way or request/response. Messages that correspond  
572 to one-way or request/response operations on a bidirectional interface use either the  
573 `scaCallbackDestination` user property (for request/response) or the `JMSReplyTo` destination (for  
574 one-way) to identify the destination to which messages are to be sent when operations are invoked on the  
575 callback interface. The use of `JMSReplyTo` for this purpose is to enable interaction with non-SCA JMS  
576 applications, as described below.

577 SCA runtimes MUST follow the behavior described in section 6.4 and its subsections when  
578 `binding.jms` is used in both the forward and callback directions [BJM60018].

579 SCA runtimes can use different bindings for forward calls and callbacks, however the behavior and  
580 requirements on messages is vendor-specific.

### 581 6.4.1 Invocation of operations on a bidirectional interface

582 For an SCA reference with a JMS binding and a bidirectional interface, when a request message is sent  
583 as part of a request/response MEP the SCA runtime MUST set the `scaCallbackDestination` user  
584 property in the message it creates to a JMS URI string, in the format defined by the IETF URI Scheme for  
585 Java™ Message Service 1.0 [IETFJMS], that identifies the destination to which callback messages are to  
586 be sent [BJM60011].

587 For an SCA reference with a JMS binding and bidirectional interface, when a request message is sent as  
588 part of a one-way MEP the SCA runtime MUST set the destination to which callback messages are to be  
589 sent as the `JMSReplyTo` destination in the message it creates [BJM60012].

590 For an SCA reference with a JMS binding and bidirectional interface, when a request message is sent as  
591 part of a request/response MEP, the SCA runtime MUST set the `JMSReplyTo` header in the message it  
592 creates as described in section 6.2 [BJM60013].

593 For both one-way and request/response operations, the reference's callback service can be used to  
594 identify the destination to which callback messages are to be sent.

595 For an SCA reference with a JMS binding and bidirectional interface, the SCA runtime MUST identify the  
596 callback destination from the reference's callback service binding if present, or supply a suitable callback  
597 destination if not present [BJM60014].

## 598 **6.4.2 Invocation of operations on a callback interface**

599 An SCA service with a callback interface can invoke operations on that callback interface by sending  
600 messages to the destination identified by the `scaCallbackDestination` user property, the  
601 `JMSReplyTo` destination, or the destination identified by the service's callback reference JMS binding.

602 For an SCA service with a JMS binding, the *callback destination* is identified as follows, in order of  
603 priority:

- 604 • The `scaCallbackDestination` identified by an earlier request/response operation, if not null;
- 605 • the `JMSReplyTo` destination identified by an earlier one-way operation, if not null;
- 606 • the request destination of the service's callback reference JMS binding, if specified

607 For an SCA service with a JMS binding, when a callback request message is sent for either a one-way or  
608 request/response MEP, the SCA runtime **MUST** send the callback request message to the callback  
609 destination. [BJM60015].

610 For an SCA service with a JMS binding, when a callback request message is sent and no callback  
611 destination can be identified then the SCA runtime **MUST** raise an error and throw an exception to the  
612 caller of the callback operation [BJM60016].

613 For an SCA service with a JMS binding, when a callback request message is sent the SCA runtime  
614 **MUST** set the `JMSReplyTo` destination in the callback request message as defined in sections 6.1 or 6.2  
615 as appropriate for the type of the callback operation invoked [BJM60017].

## 616 **6.4.3 Use of JMSReplyTo for callbacks for non-SCA JMS applications**

617 When interacting with non-SCA JMS applications, the assembler can choose to model a  
618 request/response message exchange using a bidirectional interface with a one-way operation in the  
619 forward and callback interfaces. In this case it is likely that the non-SCA JMS application does not  
620 support the use of the `scaCallbackDestination` user property. To support this, for one-way  
621 messages the `JMSReplyTo` header is used to identify the destination to be used to deliver callback  
622 messages, as described in sections 6.4.1 and 6.4.2.

---

## 623 7 Examples

624 The following snippets show the `sca.composite` file for the `MyValueComposite` file containing the  
625 `service` element for the `MyValueService` and a `reference` element for the `StockQuoteService`. Both  
626 the service and the reference use a JMS binding.

### 627 7.1 Minimal Binding Example

628 Snippet 7-1 shows the JMS binding being used with no further attributes or elements. In this case, it is  
629 left to the deployer to identify the resources to which the binding is connected.

```
630 <?xml version="1.0" encoding="UTF-8"?>
631 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
632     name="MyValueComposite">
633
634     <service name="MyValueService">
635         <interface.java interface="services.myvalue.MyValueService"/>
636         <binding.jms/>
637     </service>
638
639     <reference name="StockQuoteService">
640         <interface.java interface="services.stockquote.StockQuoteService"/>
641         <binding.jms/>
642     </reference>
643 </composite>
```

644 *Snippet 7-1: Minimal Binding Example*

### 645 7.2 URI Binding Example

646 Snippet 7-2 shows the JMS binding using the `@uri` attribute to specify the connection type and its  
647 information:

```
648 <?xml version="1.0" encoding="UTF-8"?>
649 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
650     name="MyValueComposite">
651
652     <service name="MyValueService">
653         <interface.java interface="services.myvalue.MyValueService"/>
654         <binding.jms uri="jms:MyValueServiceQueue?
655             activationSpecName=MyValueServiceAS&
656             ... "/>
657     </service>
658
659     <reference name="StockQuoteService">
660         <interface.java interface="services.stockquote.StockQuoteService"/>
661         <binding.jms uri="jms:StockQuoteServiceQueue?
662             connectionFactoryName=StockQuoteServiceQCF&
663             deliveryMode=1&
664             ... "/>
665     </reference>
666 </composite>
```

667 *Snippet 7-2: Binding Example with URI Specified*

### 668 7.3 Binding with Existing Resources Example

669 Snippet 7-3 shows the JMS binding using existing resources:

```
670 <?xml version="1.0" encoding="UTF-8"?>
671 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912">
```

```

672         name="MyValueComposite">
673
674         <service name="MyValueService">
675             <interface.java interface="services.myvalue.MyValueService"/>
676             <binding.jms>
677                 <destination jndiName="MyValueServiceQ" create="never"/>
678                 <activationSpec jndiName="MyValueServiceAS" create="never"/>
679             </binding.jms>
680         </service>
681 </composite>

```

682 *Snippet 7-3: Binding Example Using Existing Resources*

## 683 7.4 Resource Creation Example

684 Snippet 7-4 shows the JMS binding providing information to create JMS resources rather than using  
685 existing ones:

```

686 <?xml version="1.0" encoding="UTF-8"?>
687 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
688     name="MyValueComposite">
689
690     <service name="MyValueService">
691         <interface.java interface="services.myvalue.MyValueService"/>
692         <binding.jms>
693             <destination jndiName="MyValueServiceQueue" create="always">
694                 <property name="prop1">XYZ</property>
695                 <property name="destName">MyValueDest</property>
696             </destination>
697             <activationSpec jndiName="MyValueServiceAS" create="always"/>
698             <resourceAdapter jndiName="com.example.JMSRA"/>
699         </binding.jms>
700     </service>
701
702     <reference name="StockQuoteService">
703         <interface.java interface="services.stockquote.StockQuoteService"/>
704         <binding.jms>
705             <destination jndiName="StockQuoteServiceQueue"/>
706             <connectionFactory jndiName="StockQuoteServiceQCF"/>
707             <resourceAdapter name="com.example.JMSRA"/>
708         </binding.jms>
709     </reference>
710 </composite>

```

711 *Snippet 7-4: Binding Example that Creates a Resource*

## 712 7.5 Request/Response Example

713 Snippet 7-5 shows the JMS binding using existing resources to support request/response operations.  
714 The service uses the `JMSReplyTo` destination to send response messages, and does not specify a  
715 response queue:

```

716 <?xml version="1.0" encoding="UTF-8"?>
717 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
718     name="MyValueComposite">
719
720     <service name="MyValueService">
721         <interface.java interface="services.myvalue.MyValueService"/>
722         <binding.jms correlationScheme="sca:messageID">
723             <destination jndiName="MyValueServiceQ" create="never"/>
724             <activationSpec jndiName="MyValueServiceAS" create="never"/>
725         </binding.jms>
726     </service>
727
728     <reference name="StockQuoteService">

```

```

729     <interface.java interface="services.stockquote.StockQuoteService"/>
730     <binding.jms correlationScheme="sca:messageID">
731         <destination jndiName="StockQuoteServiceQueue"/>
732         <connectionFactory jndiName="StockQuoteServiceQCF"/>
733         <response>
734             <destination jndiName="MyValueResponseQueue"/>
735             <activationSpec jndiName="MyValueResponseAS"/>
736         </response>
737     </binding.jms>
738 </reference>
739 </composite>

```

740 *Snippet 7-5: Binding Example with a Response*

## 741 7.6 Subscription with Selector Example

742 Snippet 7-6 shows how the JMS binding is used in order to consume messages from existing JMS  
743 infrastructure. The JMS binding subscribes using selector:

```

744 <?xml version="1.0" encoding="UTF-8"?>
745 <composite xmlns="http://docs.oasis-open.org/ns/opencsa/sca/200912"
746     name="MyValueComposite">
747     <service name="MyValueService">
748         <interface.java interface="services.myvalue.MyValueService"/>
749         <binding.jms>
750             <destination jndiName="MyValueServiceTopic" create="never"/>
751             <connectionFactory jndiName="StockQuoteServiceTCF"
752                 create="never"/>
753             <messageSelection selector="Price>1000"/>
754         </binding.jms>
755     </service>
756 </composite>

```

757 *Snippet 7-6: Binding Example with a Selector*

## 758 7.7 Policy Set Example

759 A policy set defines the manner in which intents map to JMS binding properties. Snippet 7-7 illustrates an  
760 example of a policy set that defines values for the @priority attribute using the "priority" intent, and  
761 also allows setting of a value for a user JMS property using the "log" intent.

```

762 <policySet name="JMSPolicy"
763     provides="priority log"
764     appliesTo="binding.jms">
765
766     <intentMap provides="priority" default="medium">
767         <qualifier name="high">
768             <headers priority="9"/>
769         </qualifier>
770         <qualifier name="medium">
771             <headers priority="4"/>
772         </qualifier>
773         <qualifier name="low">
774             <headers priority="0"/>
775         </qualifier>
776     </intentMap>
777
778     <intentMap provides="log">
779         <qualifier>
780             <headers>
781                 <property name="user_example_log">logged</property>
782             </headers>
783         </qualifier>
784     </intentMap>
785 </policySet>

```

786 *Snippet 7-7: Example Policy Set*

787 Given the policy set in Snippet 7-7, the intents can be required on a service or reference as shown in  
788 Snippet 7-8:

```
789 <reference name="StockQuoteService" requires="priority.high log">  
790   <interface.java interface="services.stockquote.StockQuoteService"/>  
791   <binding.jms>  
792     <destination name="StockQuoteServiceQueue"/>  
793     <connectionFactory name="StockQuoteServiceQCF"/>  
794   </binding.jms>  
795 </reference>
```

796 *Snippet 7-8: Binding Example with Intents*



---

## 797 8 Conformance

798 The XML schema pointed to by the RDDDL document at the namespace URI, defined by this specification,  
799 are considered to be authoritative and take precedence over the XML schema defined in the appendix of  
800 this document. There are two categories of artifacts for which this specification defines conformance:

- 801 a) SCA JMS Binding XML Document
- 802 b) SCA Runtime

### 803 8.1 SCA JMS Binding XML Document

804 An SCA JMS Binding XML document is an SCA Composite Document or an SCA ComponentType  
805 Document, as defined by the [SCA Assembly Specification \[SCA-Assembly\]](#) Section 13.1 that uses the  
806 `binding.jms` element.

807 An SCA JMS Binding XML document MUST be a conformant SCA Composite Document or an SCA  
808 ComponentType Document, as defined by the [SCA Assembly Specification \[SCA-Assembly\]](#), and MUST  
809 comply with all statements in Appendix B: "Conformance Items" related to elements and attributes in an  
810 SCA JMS Binding XML document, notably all "MUST" statements have to be implemented.

### 811 8.2 SCA Runtime

812 An implementation that claims to conform to the requirements of an SCA Runtime defined in this  
813 specification has to meet the following conditions:

- 814 1. The implementation MUST comply with all statements in Appendix B: "Conformance Items"  
815 related to an SCA Runtime, notably all "MUST" statements have to be implemented
- 816 2. The implementation MUST conform to the [SCA Assembly Model Specification Version 1.1 \[SCA-  
817 Assembly\]](#), and to the [SCA Policy Framework Version 1.1 \[SCA-Policy\]](#)
- 818 3. The implementation MUST reject an SCA JMS Binding XML Document that is not conformant per  
819 Section 8.1

## A. JMS XML Binding Schema: sca-binding-jms-1.1.xsd

```

821 <?xml version="1.0" encoding="UTF-8"?>
822 <!-- Copyright(C) OASIS(R) 2005,2010. All Rights Reserved.
823      OASIS trademark, IPR and other policies apply. -->
824 <schema xmlns="http://www.w3.org/2001/XMLSchema"
825       targetNamespace="http://docs.oasis-open.org/ns/opencsa/sca/200912"
826       xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200912"
827       elementFormDefault="qualified">
828
829   <include schemaLocation="sca-core-1.1-cd05.xsd"/>
830
831   <complexType name="JMSBinding">
832     <complexContent>
833       <extension base="sca:Binding">
834         <sequence>
835           <element name="destination" type="sca:JMSDestination"
836             minOccurs="0"/>
837           <choice minOccurs="0" maxOccurs="1">
838             <element name="connectionFactory"
839               type="sca:JMSConnectionFactory"/>
840             <element name="activationSpec" type="sca:JMSActivationSpec"/>
841           </choice>
842           <element name="response" type="sca:JMSResponse" minOccurs="0"/>
843           <element name="headers" type="sca:JMSHeaders" minOccurs="0"/>
844           <element name="messageSelection" type="sca:JMSMessageSelection"
845             minOccurs="0"/>
846           <element name="resourceAdapter" type="sca:JMSResourceAdapter"
847             minOccurs="0"/>
848           <element name="operationProperties"
849             type="sca:JMSOperationProperties"
850             minOccurs="0" maxOccurs="unbounded"/>
851           <element ref="sca:extensions" minOccurs="0" maxOccurs="1"/>
852         </sequence>
853         <attribute name="correlationScheme" type="QName"
854           default="sca:messageID"/>
855         <attribute name="initialContextFactory" type="anyURI"/>
856         <attribute name="jndiURL" type="anyURI"/>
857       </extension>
858     </complexContent>
859   </complexType>
860
861   <simpleType name="JMSCreateResource">
862     <restriction base="string">
863       <enumeration value="always"/>
864       <enumeration value="never"/>
865       <enumeration value="ifNotExist"/>
866     </restriction>
867   </simpleType>
868
869   <complexType name="JMSDestination">
870     <sequence>
871       <element name="property" type="sca:BindingProperty"
872         minOccurs="0" maxOccurs="unbounded"/>
873     </sequence>
874     <attribute name="jndiName" type="anyURI"/>
875     <attribute name="type" use="optional" default="queue">
876       <simpleType>
877         <restriction base="string">
878           <enumeration value="queue"/>
879           <enumeration value="topic"/>

```

```

880     </restriction>
881   </simpleType>
882 </attribute>
883   <attribute name="create" type="sca:JMSCreateResource"
884     use="optional" default="ifNotExist"/>
885 </complexType>
886
887 <complexType name="JMSConnectionFactory">
888   <sequence>
889     <element name="property" type="sca:BindingProperty"
890       minOccurs="0" maxOccurs="unbounded"/>
891   </sequence>
892   <attribute name="jndiName" type="anyURI"/>
893   <attribute name="create" type="sca:JMSCreateResource"
894     use="optional" default="ifNotExist"/>
895 </complexType>
896
897 <complexType name="JMSActivationSpec">
898   <sequence>
899     <element name="property" type="sca:BindingProperty"
900       minOccurs="0" maxOccurs="unbounded"/>
901   </sequence>
902   <attribute name="jndiName" type="anyURI"/>
903   <attribute name="create" type="sca:JMSCreateResource"
904     use="optional" default="ifNotExist"/>
905 </complexType>
906
907 <complexType name="JMSResponse">
908   <sequence>
909     <element ref="sca:wireFormat" minOccurs="0" maxOccurs="1"/>
910     <element name="destination" type="sca:JMSDestination" minOccurs="0"/>
911     <choice minOccurs="0">
912       <element name="connectionFactory" type="sca:JMSConnectionFactory"/>
913       <element name="activationSpec" type="sca:JMSActivationSpec"/>
914     </choice>
915   </sequence>
916 </complexType>
917
918 <complexType name="JMSHeaders">
919   <sequence>
920     <element name="property" type="sca:JMSUserProperty"
921       minOccurs="0" maxOccurs="unbounded"/>
922   </sequence>
923   <attribute name="type" type="string"/>
924   <attribute name="deliveryMode" default="persistent">
925     <simpleType>
926       <restriction base="string">
927         <enumeration value="persistent"/>
928         <enumeration value="nonpersistent"/>
929       </restriction>
930     </simpleType>
931   </attribute>
932   <attribute name="timeToLive" type="long" default="0"/>
933   <attribute name="priority" default="4">
934     <simpleType>
935       <restriction base="string">
936         <enumeration value="0"/>
937         <enumeration value="1"/>
938         <enumeration value="2"/>
939         <enumeration value="3"/>
940         <enumeration value="4"/>
941         <enumeration value="5"/>
942         <enumeration value="6"/>
943         <enumeration value="7"/>

```

```

944         <enumeration value="8"/>
945         <enumeration value="9"/>
946     </restriction>
947 </simpleType>
948 </attribute>
949 </complexType>
950
951 <complexType name="JMSMessageSelection">
952     <sequence>
953         <element name="property" type="sca:BindingProperty"
954             minOccurs="0" maxOccurs="unbounded"/>
955     </sequence>
956     <attribute name="selector" type="string"/>
957 </complexType>
958
959 <complexType name="JMSResourceAdapter">
960     <sequence>
961         <element name="property" type="sca:BindingProperty"
962             minOccurs="0" maxOccurs="unbounded"/>
963     </sequence>
964     <attribute name="name" type="string" use="required"/>
965 </complexType>
966
967 <complexType name="JMSOperationProperties">
968     <sequence>
969         <element name="property" type="sca:BindingProperty"
970             minOccurs="0" maxOccurs="unbounded"/>
971         <element name="headers" type="sca:JMSHeaders" minOccurs="0"/>
972     </sequence>
973     <attribute name="name" type="string" use="required"/>
974     <attribute name="selectedOperation" type="string"/>
975 </complexType>
976
977 <complexType name="BindingProperty">
978     <simpleContent>
979         <extension base="string">
980             <attribute name="name" type="NMTOKEN" use="required"/>
981             <attribute name="type" type="string" use="optional"
982                 default="xs:string"/>
983         </extension>
984     </simpleContent>
985 </complexType>
986
987 <simpleType name="JMSUserPropertyType">
988     <restriction base="string">
989         <enumeration value="boolean"/>
990         <enumeration value="byte"/>
991         <enumeration value="short"/>
992         <enumeration value="int"/>
993         <enumeration value="long"/>
994         <enumeration value="float"/>
995         <enumeration value="double"/>
996         <enumeration value="String"/>
997         <enumeration value="xs:string"/>
998     </restriction>
999 </simpleType>
1000
1001 <complexType name="JMSUserProperty">
1002     <simpleContent>
1003         <extension base="string">
1004             <attribute name="name" type="NMTOKEN" use="required"/>
1005             <attribute name="type" type="sca:JMSUserPropertyType"
1006                 use="optional" default="String"/>
1007         </extension>

```

```
1008     </simpleContent>
1009 </complexType>
1010
1011 <complexType name="JMSDefaultWireFormatType">
1012   <complexContent>
1013     <extension base="sca:WireFormatType"/>
1014   </complexContent>
1015 </complexType>
1016
1017 <complexType name="JMSDefaultOperationSelectorType">
1018   <complexContent>
1019     <extension base="sca:OperationSelectorType"/>
1020   </complexContent>
1021 </complexType>
1022
1023 <element name="binding.jms" type="sca:JMSBinding"
1024   substitutionGroup="sca:binding"/>
1025
1026 <element name="wireFormat.jmsDefault"
1027   type="sca:JMSDefaultWireFormatType"
1028   substitutionGroup="sca:wireFormat"/>
1029
1030 <element name="operationSelector.jmsDefault"
1031   type="sca:JMSDefaultOperationSelectorType"
1032   substitutionGroup="sca:operationSelector"/>
1033 </schema>
```

1034

## B. Conformance Items

1035

This section contains a list of conformance items for the SCA JMS Binding specification.

Conformance ID	Description
[BJM30001]	The value of the @uri attribute MUST have the format defined by the <a href="#">IETF URI Scheme for Java™ Message Service 1.0 [IETFJMS]</a>
[BJM30002]	When the @uri attribute is specified, the SCA runtime MUST raise an error if the referenced resources do not already exist
[BJM30003]	If the value of the @correlationScheme attribute is "sca:messageID" the SCA runtime MUST set the correlation ID of replies to the message ID of the corresponding request
[BJM30004]	If the value of the @correlationScheme attribute is "sca:correlationID" the SCA runtime MUST set the correlation ID of replies to the correlation ID of the corresponding request
[BJM30005]	If the value of the @correlationScheme attribute is "sca:none" the SCA runtime MUST NOT set the correlation ID in responses that it sends
[BJM30006]	SCA runtimes MAY allow other values of the @correlationScheme attribute to indicate other correlation schemes
[BJM30007]	If the value of the @correlationScheme attribute is "sca:correlationID" the SCA runtime MUST set a non-null correlation ID value in requests that it sends
[BJM30010]	Whatever the value of the destination/@type attribute, the SCA runtime MUST ensure a single response is delivered for request/response operations
[BJM30011]	If the @create attribute value for a destination, connectionFactory or activationSpec element is "always" and the @jndiName attribute is present and the resource cannot be created at the location specified by the @jndiName attribute then the SCA runtime MUST raise an error
[BJM30012]	If the @create attribute value for a destination, connectionFactory or activationSpec element is "ifNotExist" then the @jndiName attribute MUST specify the location of the possibly existing resource
[BJM30013]	If the @create attribute value for a destination, connectionFactory or activationSpec element is "ifNotExist" and the resource does not exist at the location identified by the @jndiName attribute and cannot be created there then the SCA runtime MUST raise an error
[BJM30014]	If the @create attribute value for a destination, connectionFactory or activationSpec element is "ifNotExist" and the @jndiName attribute refers to an existing resource that is not a JMS Destination of the appropriate type, a JMS connection factory or a JMS activation spec respectively then the SCA runtime MUST raise an error
[BJM30015]	If the @create attribute value for a destination, connectionFactory or activationSpec element is "never" and the @jndiName attribute is not specified, or the resource is not present at the location identified by the

	@jndiName attribute, or the location refers to a resource of an incorrect type then the SCA runtime MUST raise an error
[BJM30017]	A binding.jms element MUST NOT include both a connectionFactory element and an activationSpec element
[BJM30018]	When the connectionFactory element is present as a child of the binding.jms element, then the destination MUST be defined either by the destination element child of the binding.jms element or the @uri attribute of the binding.jms element
[BJM30019]	If the activationSpec element is present as a child of the binding.jms element and the destination is also specified via a destination element child of the binding.jms element or the @uri attribute of the binding.jms element then it MUST refer to the same JMS destination as the activationSpec
[BJM30020]	The activationSpec element MUST NOT be present when the binding is being used for an SCA reference
[BJM30021]	A response element MUST NOT include both a connectionFactory element and an activationSpec element
[BJM30022]	If a response/destination and response/activationSpec element are both specified they MUST refer to the same JMS destination
[BJM30023]	The response/activationSpec element MUST NOT be present when the binding is being used for an SCA service
[BJM30024]	<p>When sending messages for a JMS binding, the SCA runtime MUST set each of the JMSType, JMSDeliveryMode, JMSTimeToLive and JMSPriority headers to values specified in the binding definition in the following priority order:</p> <ol style="list-style-type: none"> <li>1) the value for the header specified in the @uri attribute (highest priority);</li> <li>2) the value for the header specified in the operationProperties/headers element matching the operation being invoked;</li> <li>3) the value for the header specified in the headers element;</li> <li>4) the default value for the header as specified by the definition of the binding.jms/headers element (lowest priority)</li> </ol>
[BJM30025]	<p>When sending messages for a JMS binding, the SCA runtime MUST set each named user property with type and value specified in the binding definition in the following priority order:</p> <ol style="list-style-type: none"> <li>1) the type and value for the named user property specified in an operationProperties/headers/property element matching the name of the operation being invoked (highest priority);</li> <li>2) the type and value for the named user property specified in a headers/property element (lowest priority)</li> </ol>
[BJM30026]	<p>When receiving messages for a JMS binding, the SCA runtime MUST use a message selector if specified in the binding definition in the following priority order:</p> <ol style="list-style-type: none"> <li>1) the value for the message selector specified in the @uri attribute value's "selector" parameter (highest priority);</li> <li>2) the value for the message selector specified in the</li> </ol>

	messageSelection/@selector attribute; 3) otherwise no message selector is used (lowest priority)
[BJM30028]	<del>SCA runtimes MAY place restrictions on the properties of the resource adapter Java bean that can be set using the resourceAdapter element</del>
[BJM30029]	The value of the <b>operationProperties/@selectedOperation</b> attribute MUST be unique across the containing binding.jms element
[BJM30030]	<del>The SCA runtime SHOULD make the operationProperties element corresponding to the selectedOperation available to the wireFormat implementation</del>
[BJM30031]	The resourceAdapter element MUST be present when JMS resources are to be created for a JMS provider that implements the JCA 1.5 Specification [JCA15] specification, and is ignored otherwise
[BJM30034]	When the @uri attribute is specified, the destination element MUST NOT be present
[BJM30036]	The binding.jms element MUST conform to the XML schema defined in sca-binding-jms-1.1.xsd
[BJM30037]	If the @create attribute value for a destination, connectionFactory or activationSpec element is "always" and the @jndiName attribute is not present and the resource cannot be created, then the SCA runtime MUST raise an error
[BJM40001]	<del>The SCA runtime MUST support the default JMS wire format and operation selector behavior, and MAY provide additional means to override it</del>
[BJM40002]	If no operationSelector element is specified then SCA runtimes MUST use operationSelector.jmsDefault as the default
[BJM40003]	When using the default wire format to send request messages, if there is a single parameter and the interface includes more than one operation, the SCA runtime MUST set the JMS user property "scaOperationName" to the name of the operation being invoked
[BJM40004]	If no wireFormat element is specified in a JMS binding then SCA runtimes MUST use wireFormat.jmsDefault as the default
[BJM40005]	When using the default wire format an SCA runtime MUST be able to receive both JMS text and bytes messages
[BJM40006]	When using the default wire format an SCA runtime MUST send either a JMS text or a JMS bytes message
[BJM40007]	<del>When using the default wire format an SCA runtime MAY provide additional configuration to allow selection between JMS text or bytes messages to be sent</del>
[BJM40008]	When a binding.jms element specifies the operationSelector.jmsDefault element, the SCA runtime MUST use the default operation selection algorithm to determine the selected operation
[BJM40009]	When a binding.jms element specifies the wireFormat.jmsDefault element, the SCA runtime MUST use the default wire format



[BJM40010]	When a message is received at an SCA service with JMS binding and the resolved operation name is in the target component's interface, the SCA runtime MUST invoke the target component using the resolved operation name
[BJM40011]	When a message is received at an SCA service with JMS binding and the resolved operation name is not in the target component's interface the SCA runtime MUST raise an error
[BJM50001]	JMS binding implementations MUST support the JMS intent
[BJM50002]	The JMS intent MUST always be included in the @alwaysProvides attribute of the JMS bindingType
[BJM60004]	<del>For an SCA reference with a JMS binding and unidirectional interface, when a request message is sent as part of a one-way MEP, the SCA runtime SHOULD NOT set the JMSReplyTo destination header in the JMS message that it creates, regardless of whether the JMS binding has a response element with a destination defined</del>
[BJM60002]	For an SCA service with a JMS binding and unidirectional interface, when a request message is received as part of a one-way MEP, the SCA runtime MUST ignore the JMSReplyTo destination header in the JMS message, and not raise an error
[BJM60003]	For an SCA reference with a JMS binding that has a destination specified via the response element, the SCA runtime MUST receive response messages as defined by the binding's @correlationScheme attribute
[BJM60004]	For an SCA reference with a JMS binding, when a request message is sent as part of a request/response MEP, and the JMS binding has a response element with a destination defined, then the SCA runtime MUST use that destination for the JMSReplyTo header in the JMS message it creates for the request
[BJM60005]	For an SCA reference with a JMS binding, when a request message is sent as part of a request/response MEP, and the JMS binding does not have a response element with a destination defined, the SCA runtime MUST provide an appropriate destination on which to receive response messages, and use that destination for the JMSReplyTo header in the JMS message it creates for the request
[BJM60006]	For an SCA reference with a JMS binding that does not have a destination specified via the response element, the SCA runtime MUST either receive response messages as defined by the binding's @correlationScheme attribute, or use a unique destination for each request/response interaction
[BJM60007]	For an SCA service with a JMS binding, when a response message is sent as part of a request/response MEP where the request message included a non-null JMSReplyTo destination, the SCA runtime MUST send the response message to that destination
[BJM60008]	For an SCA service with a JMS binding, when a response message is sent as part of a request/response MEP where the request message included a null JMSReplyTo destination and the JMS binding includes a response/destination element the SCA runtime MUST send the response message to that destination

[BJM60009]	For an SCA service with a JMS binding, when a <code>responseRequest</code> message is <del>sent</del> <code>received</code> as part of a request/response MEP where the request message includes a null <code>JMSReplyTo</code> destination and the JMS binding does not include a response/destination then <del>an error SHOULD be raised by the SCA runtime</del> <b>MUST NOT process the request and MUST raise an error</b>
[BJM60010]	For an SCA service with a JMS binding, when a response message is sent as part of a request/response MEP the SCA runtime <b>MUST</b> set the correlation identifier in the JMS message that it creates for the response as defined by the JMS binding's <code>@correlationScheme</code> attribute
[BJM60011]	For an SCA reference with a JMS binding and a bidirectional interface, when a request message is sent as part of a request/response MEP the SCA runtime <b>MUST</b> set the <code>scaCallbackDestination</code> user property in the message it creates to a JMS URI string, in the format defined by the IETF URI Scheme for Java™ Message Service 1.0 [IETF JMS], that identifies the destination to which callback messages are to be sent
[BJM60012]	For an SCA reference with a JMS binding and bidirectional interface, when a request message is sent as part of a one-way MEP the SCA runtime <b>MUST</b> set the destination to which callback messages are to be sent as the <code>JMSReplyTo</code> destination in the message it creates
[BJM60013]	For an SCA reference with a JMS binding and bidirectional interface, when a request message is sent as part of a request/response MEP, the SCA runtime <b>MUST</b> set the <code>JMSReplyTo</code> header in the message it creates as described in section 6.2
[BJM60014]	For an SCA reference with a JMS binding and bidirectional interface, the SCA runtime <b>MUST</b> identify the callback destination from the reference's callback service binding if present, or supply a suitable callback destination if not present
[BJM60015]	For an SCA service with a JMS binding, when a callback request message is sent for either a one-way or request/response MEP, the SCA runtime <b>MUST</b> send the callback request message to the callback destination.
[BJM60016]	For an SCA service with a JMS binding, when a callback request message is sent and no callback destination can be identified then the SCA runtime <del>SHOULD</del> <b>MUST</b> raise an error, and <del>MUST</del> throw an exception to the caller of the callback operation
[BJM60017]	For an SCA service with a JMS binding, when a callback request message is sent the SCA runtime <b>MUST</b> set the <code>JMSReplyTo</code> destination in the callback request message as defined in sections 6.1 or 6.2 as appropriate for the type of the callback operation invoked
[BJM60018]	SCA runtimes <b>MUST</b> follow the behavior described in section 6.4 and its subsections when <code>binding.jms</code> is used in both the forward and callback directions

1037

## C. Acknowledgements

1038 The following individuals have participated in the creation of this specification and are gratefully  
1039 acknowledged:

1040 **Participants:**

Participant Name	Affiliation
Bryan Aupperle	IBM
Ron Barack	SAP AG
Michael Beisiegel	IBM
Henning Blohm	SAP AG
David Booz	IBM
Martin Chapman	Oracle Corporation
Jean-Sebastien Delfino	IBM
Laurent Domenech	TIBCO Software Inc.
Jacques Durand	Fujitsu Limited
Mike Edwards	IBM
Billy Feng	Primeton Technologies, Inc.
Nimish Hathalia	TIBCO Software Inc.
Simon Holdsworth	IBM
Eric Johnson	TIBCO Software Inc.
Uday Joshi	Oracle Corporation
Khanderao Kand	Oracle Corporation
Anish Karmarkar	Oracle Corporation
Nickolaos Kavantzias	Oracle Corporation
Mark Little	Red Hat
Ashok Malhotra	Oracle Corporation
Jim Marino	Individual
Jeff Mischkinisky	Oracle Corporation
Dale Moberg	Axway Software
Simon Nash	Individual
Sanjay Patil	SAP AG
Plamen Pavlov	SAP AG
Peter Peshev	SAP AG
Piotr Przybylski	IBM
Luciano Resende	IBM
Tom Rutt	Fujitsu Limited
Vladimir Savchenko	SAP AG
Scott Vorthmann	TIBCO Software Inc.
Tim Watson	Oracle Corporation
Owen Williams	Avaya, Inc.
Prasad Yendluri	Software AG, Inc.

1041

## D. Revision History

1042

Revision	Date	Editor	Changes Made
1	2007-09-25	Anish Karmarkar	Applied the OASIS template + related changes to the Submission
2	2008-03-12	Simon Holdsworth	Updated text for RFC2119 conformance Updates to resolve following issues: BINDINGS-1 BINDINGS-5 BINDINGS-6 BINDINGS-12 BINDINGS-14 BINDINGS-18 BINDINGS-26 Applied updates discussed at Bindings TC meeting of 27 <sup>th</sup> March
3	2008-06-19	Simon Holdsworth	* Applied most of the editorial changes from Eric Johnson's review
cd01	2008-08-01	Simon Holdsworth	Updates to resolve following issues: BINDINGS-13 (JMS part) BINDINGS-20 (complete) BINDINGS-30 (JMS part) BINDINGS-32 (JMS part) BINDINGS-33 (complete) BINDINGS-34 (complete) BINDINGS-35 (complete) BINDINGS-38 (JMS part)
cd01-rev1	2008-10-16	Simon Holdsworth	Updated text for RFC2119 conformance throughout Updates to resolve following issues: BINDINGS-41 BINDINGS-46 BINDINGS-47
cd01-rev2	2008-12-01	Simon Holdsworth	Added comments identifying those updates that relate to RFC2119 language (issue 52)
cd01-rev3	2008-12-02	Simon Holdsworth	Final RFC2119 language updates BINDINGS-52
cd01-rev4	2009-01-09	Simon Holdsworth	Updates to resolve following issues:

			BINDINGS-7 BINDINGS-31 BINDINGS-40 BINDINGS-42 BINDINGS-44 BINDINGS-50
cd02	2009-02-16	Simon Holdsworth	Rename and editorial updates
cd02-rev1	2009-05-22	Simon Holdsworth	Updates to resolve issue BINDINGS-62 (conformance statement numbering) Updated assembly namespace to 200903 Fixed errors in schema
cd02-rev2	2009-05-22	Simon Holdsworth	Updates to resolve following issues: BINDINGS-39 BINDINGS-59 BINDINGS-65 BINDINGS-66 BINDINGS-67 BINDINGS-68 BINDINGS-70 BINDINGS-71
cd02-rev3	2009-06-18	Simon Holdsworth	Editorial concerns addressed Added acknowledgements appendix
cd02-rev4	2009-06-19	Simon Holdsworth	Updates to resolve following issues BINDINGS-74 Some editorial updates Fixed normative statement missed in application of BINDINGS-67
cd02-rev5	2009-06-24	Simon Holdsworth	Updates to resolve following issues BINDINGS-77 Renamed document to old form Removed editorial commentary Editorial fixes around external references; changed all links to hyperlinks
cd02-rev6	2009-06-24	Simon Holdsworth	Fixed application of BINDINGS-74 Fixed broken cross reference Changed ASCII to UTF-8 in examples
cd03	2009-06-29	Simon Holdsworth	Updates to resolve following issues BINDINGS-80 BINDINGS-81
cd03-rev1	2010-01-24	Simon Holdsworth	Editorial fix to XML schema name

			Updated to resolve following issues BINDINGS-48 BINDINGS-83 BINDINGS-85 BINDINGS-90 BINDINGS-93 BINDINGS-94 BINDINGS-96 BINDINGS-97 BINDINGS-98 BINDINGS-103 BINDINGS-108 BINDINGS-109 BINDINGS-110
cd03-rev2	2010-02-12	Simon Holdsworth	Editorial fixes to cross-references Fix cd03-rev1 change to add BINDINGS-110 Updated to resolve following issues BINDINGS-95 BINDINGS-104 BINDINGS-105 BINDINGS-106
cd03-rev3	2010-02-17	Bryan Aupperle	Add captions to all diagrams
cd03-rev4	2010-02-22	Simon Holdsworth	Updated assembly namespace to 200912 Editorial updates from action items and issues BINDINGS-101 BINDINGS-102 20091015-3: no change to copyright (currently consistent with all other SCA specs) 20091015-8: removed non-normative references section 20091015-9: cleaned up naming conventions section 20091015-10: cleaned up some phrases that used "may" or "allows" 20091015-12: no changes made (currently consistent with all other SCA specs)
cd03-rev5	2010-03-18	Simon Holdsworth	Fixed application of issue BINDINGS-108 Editorial cleanup Changed assembly reference to CD05
cd03-rev6	2010-04-16	Simon Holdsworth	Applied resolution to BINDINGS-128
cd04	2010-04-30	Simon Holdsworth	Rename and fix acknowledgements, fixup for publication

cd04-rev1	2010-10-05	Simon Holdsworth	Applied resolutions for issues: BINDINGS-134 BINDINGS-135 BINDINGS-136 BINDINGS-138 BINDINGS-139 Updated SCA policy spec reference and IETF JMS URI draft reference
cd04-rev2	2010-10-26	Simon Holdsworth	Applied resolutions for issues: BINDINGS-141
cd04-rev3	2010-10-29	Simon Holdsworth	Applied resolutions for issues: BINDINGS-140
<a href="#">csprd03-rev1</a>	<a href="#">2011-05-22</a>	<a href="#">Simon Holdsworth</a>	<a href="#">Applied resolutions for issues:</a> <a href="#">BINDINGS-144</a> <a href="#">BINDINGS-149</a> <a href="#">BINDINGS-154</a> <a href="#">BINDINGS-156</a> <a href="#">BINDINGS-157</a> <a href="#">BINDINGS-158</a> <a href="#">BINDINGS-159</a>
<a href="#">csdprd03-rev2</a>	<a href="#">2011-07-04</a>	<a href="#">Simon Holdsworth</a>	<a href="#">Applied resolutions for issues:</a> <a href="#">BINDINGS-169</a>

1043