

SCA Service Component Architecture

JCA Binding Specification

SCA Version 1.00 20 September 2007

Technical Contacts:	Bryan Aupperle	IBM Corporation
	Martin Chapman	Oracle
	Codanda Chinnappa	BEA Systems, Inc.
	Eric Johnson	TIBCO Software Inc.
	Peter Peshev	SAP AG
	Piotr Przybylski	IBM Corporation
	Michael Rowley	BEA Systems, Inc.

Copyright Notice

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

License

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

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

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

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

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

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

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

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

Status of this Document

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

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

BEA is a registered trademark of BEA Systems, Inc.

Cape Clear is a registered trademark of Cape Clear Software

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

Oracle is a registered trademark of Oracle USA, Inc.

Progress is a registered trademark of Progress Software Corporation

Red Hat is a registered trademark of Red Hat Inc.

Rogue Wave is a registered trademark of Rogue Wave Software, Inc

SAP is a registered trademark of SAP AG.

SIEMENS is a registered trademark of SIEMENS AG

Software AG is a registered trademark of Software AG

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

Sybase is a registered trademark of Sybase, Inc.

TIBCO is a registered trademark of TIBCO Software, Inc.

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

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

Table of Contents

SCA Service Component Architecture.....	i
Copyright Notice.....	ii
License	ii
Status of this Document.....	iii
1 JCA Bindings	1
1.1 Introduction	1
1.2 Operation Selector and Data Bindings.....	1
1.3 JCA Binding.....	2
1.4 Binding Properties.....	5
1.5 Additional binding configuration data.....	7
1.6 Examples.....	7
1.6.1 Minimal JCA Binding	7
1.6.2 Existing resources.....	7
1.6.3 Resource Creation.....	8
1.6.4 Existing Resources specified in the definition file.....	9
1.7 TODO – Open Issues	9
2 JCA Binding Schema.....	10
Appendix 1.....	16
JavaEE Connector Architecture Introduction.....	16
Selected JCA CCI Interfaces.....	17
Record	17
Interaction.....	17
MessageListener	17
Appendix 2 - References	18

1 JCA Bindings

1.1 Introduction

This document presents bindings describing access and connectivity to the services provided by the Enterprise Information System (EIS).

This version of the document describes JCA Bindings thus narrowing connectivity down to the connectivity to the EIS system external to the SCA system, based on the J2EE Connector Architecture specification and implemented in Java.

Further specification is necessary to define EIS Bindings between different SCA runtimes within SCA system, for example J2EE and EIS based runtimes.

The JCA Bindings are applicable to the composite's references and services.

The connection to exchange data with the EIS is characterized by two sets of configuration parameters, the connection and interaction parameters. The former set determines the location of the target system the latter determines characteristics that need to be specified to invoke one specific service available at the endpoint. JCA Binding model captures these parameters as separate sets to allow their reuse and reconfiguration.

1.2 Operation Selector and Data Bindings

The Operation Selector is used to identify the operation of the service that should be invoked. The selector extracts the native function name or identifier, for example event identifier, from the data flowing inbound from the EIS.

The implementation of the Operation Selector can be provided by the binding provider or third party who understands how to locate, in the native data the appropriate identifier, for example by the JCA Resource Adapter provider.

The data bindings are responsible for data format conversion between EIS and runtime format of data. They can be provided by the EIS API vendor, the runtime implementation or third party. The data bindings can be specific to the EIS system or to the data format (e.g. XML).

This specification does not define default behavior for the Operation Selector or Data Binding functionality. This choice had been made because the interfaces describing data exchanged between JCA adapter and its client are specific to a particular adapter and, unlike JMS, cannot be used in a generic manner.

The JCA Binding implementation however, needs to use or provide the Operation Selection and Data Binding functionality. To support multiple adapters in a generic fashion, the binding needs to expose a mechanism for selecting or specifying the right implementations of data bindings or operation selector a protocol that must be followed by this functionality providers. This can be achieved in a variety of ways, for example by providing the metadata information or specific pluggability points. This document does not specify such mechanism, it is left to the binding implementation.

42 **1.3 JCA Binding**

43 The SCA Bindings for JCA is defined with the following schema:

```

44
45 <binding.jca connectionInfo="QName"?>
46
47 <jca.outbound.connection managed="xs:boolean"?>
48     <resourceAdapter name="NMTOKEN"? type="NMTOKEN"?>?
49         <property name="NMTOKEN" type="NMTOKEN"?*>
50     </resourceAdapter>
51     <connection name="NMTOKEN"? type="NMTOKEN" create="string"?>
52         <property name="NMTOKEN" type="NMTOKEN"?*>
53     </connection>
54     <resAuth>Container|Application</resAuth>?
55     <!-- Vendor specific extensions -->
56 </jca.outbound.connection>?
57
58 <jca.inbound.connection>
59     <resourceAdapter name="NMTOKEN"? type="NMTOKEN">
60         <property name="NMTOKEN" type="NMTOKEN"?*>
61     </resourceAdapter>
62     <activationSpec name="NMTOKEN"? type="NMTOKEN" create="string"?>
63         <property name="NMTOKEN" type="NMTOKEN"?*>
64     </activationSpec>
65     <!-- Vendor specific extensions -->
66 </jca.inbound.connection>?
67
68 <jca.outbound.interaction>
69     <connectionSpec type="NMTOKEN" >?
70         <property name="NMTOKEN" type="NMTOKEN"?*>
71 </connectionSpec>
72     <interactionSpec type="NMTOKEN" >?
73         <property name="NMTOKEN" type="NMTOKEN"?*>
74     </interactionSpec>
75     <operation name="NMTOKEN"?*>
76     <interactionSpec type="NMTOKEN"?>?
77         <property name="NMTOKEN" type="NMTOKEN"?*>

```

```

78         </interactionSpec>
79     </operation>
80     <!-- Vendor specific extensions -->
81     </jca.outbound.interaction>?
82 <jca.inbound.interaction>
83     <listener type="NMTOKEN"/>?
84     <inboundOperation name="NMTOKEN" nativeOperation="NMTOKEN">*
85     </inboundOperation>
86
87     </jca.inbound.interaction>?
88
89 </binding.jca>

```

- 92 • The **uri** element (from the **binding**) allows for the specification of the endpoint. For the
93 reference, the **uri** element defines the endpoint allowing connecting to the target EIS by
94 providing JNDI name under which the ConnectionFactory is located. For the service, the **uri**
95 defines the endpoint to allow the EIS system to connect to the SCA system by defining the JNDI
96 lookup name of the ActivationSpec, for example `uri="java:comp/env/eis/TRAN_EIS"`. The uri is
97 exclusive with the connectionInfo element as well as `jca.inbound.connection` or
98 `jca.outbound.connection` elements
- 99 • The **connectionInfo** identifies the `jca.binding` element present in the definitions document and
100 whose child or children (one or more of `jca.inbound.connection`, `jca.outbound.connection`,
101 `jca.inbound.interaction`, `jca.outbound.interaction`) are used to define characteristics of connection
102 and interaction for this binding.
- 103 • The **jca.outbound.connection** defines the outbound connection characteristics and contains the
104 following:
 - 105 • The **managed** attribute that determines whether the interaction with the EIS system should be
106 performed in the managed or non-managed mode. If the value is true (default), the JNDI name
107 is used to obtain connection to the EIS and use adapter in the managed mode. If the value is
108 false, the connection information is used to invoke adapter in the non-managed mode i.e. by
109 creating instance of the ManagedConnectionFactory and using it to create Connection. For the full
110 description of the managed and non-managed mode refer to section 6.9 of the J2EE Connector
111 Architecture Specification [5].
 - 112 • The **resourceAdapter** – specifies name, type and properties of the Resource Adapter Java bean.
113 There may be a restriction, depending on the deployment platform, about specifying properties
114 of the RA Java Bean. This element is only valid in the managed mode.
 - 115 ○ **type** – the fully qualified name of the class implementing the JCA ResourceAdapter
116 interface
 - 117 ○ **name** – the optional name that uniquely identifies the existing instance of the resource
118 adapter.

- 119 • The **properties** element contains the subset of the properties of the Resource Adapter Java Bean
 120 that need to be set in order to access specified EIS service. The full list of Resource Adapter
 121 properties can be obtained by introspecting the Java Bean.
- 122 • The **connection** element specifies the properties of the connection factory used to create
 123 connections to the required service endpoint.
- 124 ○ **type** – the fully qualified name of the class implementing the JCA
 125 ManagedConnectionFactory interface
 - 126 ○ **name** – if the create attribute is never, the name uniquely identifies existing instance of
 127 the managed connection factory. If create attribute is always, the name must be unique
 128 within domain.
- 129 • The **properties** element contains the subset of the properties of the Managed Connectoin
 130 Factory Java Bean that need to be set in order to access specified EIS service. The full list of
 131 Managed Connectoin Factory properties can be obtained by introspecting the Java Bean.
- 132 ○ The **create** attribute indicates whether the element containing the attribute should be
 133 created when the containing composite is deployed. Valid values are "always", "never"
 134 and "ifnotexist". The default value is "ifnotexist". It is an error if the attribute value is
 135 "always" and the element with the given name already exists.
- 136 • The **resAuth** element specifies the authentication mechanism used by the resource adapter in
 137 the managed environment
- 138 • Vendor specific extensions allow to customize the model to support the specific runtime
 139 characteristics, for example pool size or maximum number of connections
- 140 • The **jca.outbound.interaction** defines characteristics of the outbound interaction and contains
 141 the following elements
- 142 • The **connectionSpec** identifies the name of the class implementing
 143 javax.resource.cci.ConnectionSpec interface and the set of connectionSpec properties to be
 144 specified when creating a connection, a client level connection properties e.g. user name or
 145 password. The ConnectionSpec object is used in several patterns that justify its definition in the
 146 interaction binding.
 - 147 • The **interactionSpec** type specifies the name of the class implementing
 148 javax.resource.cci.InteractionSpec interface. The interaction specified outside of all operation
 149 applies to all the operations defined
 - 150 • The **operation** element gathers characteristics of one operation of the service, the data bindings
 151 of the inbound and outbound arguments as well as interaction type and the properties.
- 152 • The **jca.inbound.connection** information side file has the following format, applicable to the
 153 managed connectivity (the JCA specification does not support inbound non-managed scenario).
- 154 • The **resourceAdapter** – specifies name, type and properties of the Resource Adapter Java bean.
 155 There may be a restriction, depending on the deployment platform, about specifying properties
 156 of the RA Java Bean. This element is only valid in the managed mode.
- 157 ○ **type** – the fully qualified name of the class implementing the ResourceAdapter interface
 - 158 ○ **name** – the optional name that uniquely identifies the existing instance of the resource
 159 adapter.
- 160 • The **activationSpec** element specifies the name of the class implementing
 161 javax.resource.spi.ActivationSpec interface and its properties.
- 162 ○ **type** – the fully qualified name of the class implementing the ActivationSpec interface

- 163 o **name** – if the create attribute is never, the name uniquely identifies existing instance of
164 the activation spec. If create attribute is always, the name must be unique within domain.
- 165 o The **create** attribute indicates whether the element containing the attribute should be
166 created when the containing composite is deployed. Valid values are "always", "never"
167 and "ifnotexist". The default value is "ifnotexist". It is an error if the attribute value is
168 "always" and the element with the given name already exists.
- 169 • The **jca.inbound.interaction** defines characteristics of the individual interaction in following
170 format
- 171 • The **listener** type specifies the listener interface supported by this group of interactions. If the
172 listener is not specified, it is assumed to be a listener implementing
173 javax.resource.cci.MessageListener interface from the JCA specification
- 174 • The **inboundOperation** element maps the name of the EIS event received by ResourceAdapter
175 to the name of the operation of the Service.

176

177 **Extensibility** - the JCA Bindings provide an extensibility mechanism that allows further
178 customization of the bindings with the vendor specific attributes or elements using extensibility
179 element in the schema as follows:

- 180 o `<anyAttribute namespace="##any" processContents="lax" />`
- 181 o `<any namespace="##other" processContents="lax" minOccurs="0"`
182 `maxOccurs="unbounded" />`

183

184 **1.4 Binding Properties**

185

186 The JCA Binding may contain properties necessary to interact with the EIS system, properties
187 that are, however, not related to the service location or type of services available. Such
188 properties should be configurable but should not require overwriting connection or interaction
189 elements. Examples of such properties are user ID or password.

190 The binding.jca contains connectionInfo element that specifies the name of the binding.jca
191 element in the definition file.

192

```
193 <reference name="EISHelloWorldReference">
194     <binding.jca uri=" java:comp/env/eis/EISMCF "
195         connectionInfo="JCA_Services">
196     </binding.jca>
197 </reference>
```

198

199 This element may contain the interaction properties, for example properties of the
200 connectionSpec.

```
201 <definitions targetNamespace="http://acme.com"
202     xmlns="http://www.oxa.org/xmlns/sca/1.0">
203     <binding.jca name="JCA_Services">
```

```

204     <jca.outbound.interaction >
205         <connectionSpec name="FAConnectionSpec">
206             <property name="group">GROUP1</property>
207             <property name="userid">SYSAD</property>
208             <property name="password">SYSAD</property>
209         </connectionSpec>
210         ...
211     </jca.outbound.interaction>
212 </binding.jca>
213 </definitions>
214

```

In the example above, the connectionSpec element specifies all the properties it overwrites in place and needs to be updated when there is a need to modify any of the properties. This could be inefficient at times and the method of passing properties from the bindings is defined. To get the value from the bindings, the property specifies the source attribute as follows.

```

219
220     <jca.outbound.interaction >
221         <connectionSpec name="connector.file.outbound.FAConnectionSpec">
222             <property name="group">GROUP1</property>
223             <property name="userid">SYSAD</property>
224             <property name="password" source="$password"/>
225         </connectionSpec>
226     </jca.outbound.interaction>
227

```

The property value is the specified in the binding element that refers to the element in the definitions file.

```

230
231     <reference name="JCAHelloWorldReference">
232         <binding.jca uri=" java:comp/env/eis/MCF "
233             connectionInfo="JCA_Services">
234             <property name="password">SYSAD</property>
235         </binding.jca>
236     </reference>
237

```

The properties can also be specified by the composite, in that case the reference or service would contain the source attribute pointing to the property of the composite:

```

241
242
243 <composite xmlns="http://www.oesa.org/xmlns/sca/1.0" name="EISHelloworld">
244
245     <reference name="EISHelloWorldReference">
246         <binding.jca uri=" java:comp/env/eis/EISMCF"
247             connectionInfo="JCA_Services">
248             <property name="userid" source="$UID"/>
249         </binding.jca>
250     </reference>
251
252
253     <property name="UID">SYSAD</property>
254 </composite>
255

```

The indirection level of the binding, required even if the property value is specified in the composite prevents introducing hidden dependencies between the composite and definitions file.

1.5 Additional binding configuration data

SCA runtime implementations may provide additional configuration that is associated with a JCA Binding, for example to overwrite binding properties like user name or password. The specification of such configuration is SCA runtime-specific and is outside of the scope of this document.

1.6 Examples

1.6.1 Minimal JCA Binding

The minimal JCA Binding only contains the URI attribute with JNDI name of the connection factory. It allows to obtain the Connection to execute request against EIS using adapter. Since no interaction properties are specified, it is assumed that Resource Adapter accepts the null values for the invocatin methods.

```

270 <!-- JCA reference, connection is configured in JNDI context -->
271 <reference name="EISHelloWorldReference">
272     <binding.jca uri="java:comp/env/eis/EISMCF"/>
273 </reference>
274

```

1.6.2 Existing resources

The sample reference with the JCA Binding, the uri specifies the existing resource - the JNDI name under which the connection factory object is located. The interaction properties are specified explicitly in the inlined jca.outbound.interaction element.

```

280 <reference name="EISHelloWorldReference">
281   <binding.jca uri="java:comp/env/eis/EISMCF">
282     <jca.outbound.interaction>
283
284       <connectionSpec name="FAConnectionSpec">
285         <property name="userid">SYSAD</property>
286       </connectionSpec>
287       <interactionSpec name="FAInteractionSpec">
288       </interactionSpec>
289
290       <operation name="hello">
291         <interactionSpec>
292           <property name="dir">temp</property>
293           <property name="fileMode">read</property>
294         </interactionSpec>
295       </operation>
296
297     </jca.outbound.interaction>
298   </binding.jca>
299 </reference>

```

1.6.3 Resource Creation

The following sample presents the reference with JCA bindings where the connection resources do not exist and need to be created.

```

304 <reference name="JCAHelloWorldReference">
305   <binding.jca>
306     <jca.outbound.connection managed="true">
307       <resourceAdapter name="connector.file.FAResourceAdapter">
308         <property name="logDrive">D</property>
309       </resourceAdapter>
310       <connection name="FAManagedConnectionFactory"
311         create="always">
312         <property name="host">localhost</property>
313         <property name="drive">C</property>
314       </connection>
315     </jca.outbound.connection>

```

```

316         </binding.jca>
317     </reference>

```

```

318
319

```

320 1.6.4 Existing Resources specified in the definition file

```
321
```

322 This sample shows the resources specified in the definitions file and referred to by the binding
 323 elements. The definitions file contains the following

```

324     <definitions targetNamespace="http://acme.com"
325         xmlns="http://www.oesa.org/xmlns/sca/1.0">
326
327         <binding.jca name="JCA_Inbound">
328             <jca.inbound.connection>
329                 <resourceAdapter name="FAResourceAdapter">
330                     <property name="logDrive">D</property>
331                 </resourceAdapter>
332                 <activationSpec name="FAActivationSpec">
333                     <property name="directory_type">temp</property>
334                     <property name="drive">C</property>
335                 </activationSpec>
336             </jca.inbound.connection>
337         </binding.jca>
338     </definitions>

```

```
339
```

340 The service with the JCA Bindings uses the connectionInfo attribute to identify the resources
 341 in the definition file

```

342     <service name="JCAHelloWorldService">
343         <binding.jca connectionInfo=" JCA_Inbound ">
344             <jca.inbound.interaction>
345                 <listener>MyInboundListener</listener>
346                 <inboundOperation name="hello" nativeOperation="TXPN"/>
347                 <inboundOperation name="bye" nativeOperation="ETXPRN"/>
348             </jca.inbound.interaction>
349         </binding.jca>
350     </service>

```

```

351
352

```

2 JCA Binding Schema

```

353 <?xml version="1.0" encoding="UTF-8"?>
354 <!-- (c) Copyright SCA Collaboration 2006, 2007 -->
355 <schema xmlns="http://www.w3.org/2001/XMLSchema"
356   targetNamespace="http://www.osoa.org/xmlns/sca/1.0"
357   xmlns:sca="http://www.osoa.org/xmlns/sca/1.0"
358   elementFormDefault="qualified">
359
360   <include schemaLocation="sca-core.xsd" />
361
362   <complexType name="JCABinding">
363     <complexContent>
364       <extension base="sca:Binding">
365         <sequence>
366           <element name="jca.outbound.connection"
367             type="sca:JCAOutboundConnection" minOccurs="0" />
368           <element name="jca.inbound.connection"
369             type="sca:JCAInboundConnection" minOccurs="0" />
370           <element name="jca.outbound.interaction"
371             type="sca:JCAOutboundInteraction" minOccurs="0" />
372           <element name="jca.inbound.interaction"
373             type="sca:JCAInboundInteraction" minOccurs="0" />
374           <element name="property" type="sca:Property"
375             minOccurs="0" maxOccurs="unbounded" />
376           <any namespace="##other" processContents="lax"
377             minOccurs="0" maxOccurs="unbounded" />
378         </sequence>
379         <attribute name="connectionInfo" type="anyURI"
380           use="optional" />
381         <anyAttribute namespace="##any" processContents="lax" />
382       </extension>
383     </complexContent>
384   </complexType>
385
386
387

```

```

388 <simpleType name="CreateResource">
389     <restriction base="string">
390         <enumeration value="always" />
391         <enumeration value="never" />
392         <enumeration value="ifnotexist" />
393     </restriction>
394 </simpleType>
395
396 <simpleType name="ResAuth">
397     <restriction base="string">
398         <enumeration value="Container" />
399         <enumeration value="Application" />
400     </restriction>
401 </simpleType>
402
403 <complexType name="JCAOutboundConnection">
404     <sequence>
405         <element name="resourceAdapter" type="sca:ResourceAdapter"
406             minOccurs="0" />
407         <element name="connection" type="sca:Connection" />
408         <element name="resAuth" type="sca:ResAuth" minOccurs="0" />
409         <any namespace="##other" processContents="lax" minOccurs="0"
410             maxOccurs="unbounded" />
411     </sequence>
412     <attribute name="managed" type="boolean" use="optional"
413         default="true" />
414     <anyAttribute namespace="##any" processContents="lax" />
415 </complexType>
416
417 <complexType name="JCAInboundConnection">
418     <sequence>
419         <element name="resourceAdapter" type="sca:ResourceAdapter" />
420         <element name="activationSpec" type="sca:ActivationSpec" />
421         <any namespace="##other" processContents="lax" minOccurs="0"
422             maxOccurs="unbounded" />
423     </sequence>

```

```

424 </complexType>
425
426 <complexType name="JCAOutboundInteraction">
427     <sequence>
428         <element name="connectionSpec" type="sca:ConnectionSpec"
429             minOccurs="0" />
430         <element name="interactionSpec" type="sca:InteractionSpec"
431             minOccurs="0" />
432         <element name="operation" type="sca:Operation"
433             minOccurs="0" />
434         <any namespace="##other" processContents="lax" minOccurs="0"
435             maxOccurs="unbounded" />
436     </sequence>
437 </complexType>
438
439 <complexType name="JCAInboundInteraction">
440     <sequence>
441         <element name="listener" type="string" minOccurs="0" />
442         <element name="inboundOperation" type="sca:InboundOperation"
443             minOccurs="0" maxOccurs="unbounded" />
444         <any namespace="##other" processContents="lax" minOccurs="0"
445             maxOccurs="unbounded" />
446     </sequence>
447 </complexType>
448
449 <complexType name="ResourceAdapter">
450     <sequence>
451         <element name="property" type="sca:Property" minOccurs="0"
452             maxOccurs="unbounded" />
453         <any namespace="##other" processContents="lax" minOccurs="0"
454             maxOccurs="unbounded" />
455     </sequence>
456     <attribute name="name" type="NMTOKEN" use="optional" />
457     <attribute name="type" type="NMTOKEN" use="required" />
458     <anyAttribute namespace="##any" processContents="lax" />
459 </complexType>

```

```

460
461 <complexType name="Connection">
462     <sequence>
463         <element name="property" type="sca:Property" minOccurs="0"
464             maxOccurs="unbounded" />
465         <any namespace="##other" processContents="lax" minOccurs="0"
466             maxOccurs="unbounded" />
467     </sequence>
468     <attribute name="name" type="NMTOKEN" use="optional" />
469     <attribute name="type" type="NMTOKEN" use="required" />
470     <attribute name="create" type="sca:CreateResource"
471         use="optional" default="ifnotexist" />
472     <anyAttribute namespace="##any" processContents="lax" />
473 </complexType>
474
475 <complexType name="ActivationSpec">
476     <sequence>
477         <element name="property" type="sca:Property" minOccurs="0"
478             maxOccurs="unbounded" />
479         <any namespace="##other" processContents="lax" minOccurs="0"
480             maxOccurs="unbounded" />
481     </sequence>
482     <attribute name="name" type="NMTOKEN" use="optional" />
483     <attribute name="type" type="NMTOKEN" use="required" />
484     <attribute name="create" type="sca:CreateResource"
485         use="optional" default="ifnotexist" />
486     <anyAttribute namespace="##any" processContents="lax" />
487 </complexType>
488
489 <complexType name="Operation">
490     <sequence>
491         <element name="interactionSpec" type="sca:InteractionSpec"
492             minOccurs="0" />
493         <any namespace="##other" processContents="lax" minOccurs="0"
494             maxOccurs="unbounded" />
495     </sequence>

```

```

496         <attribute name="name" type="NMTOKEN" use="required" />
497         <anyAttribute namespace="##any" processContents="lax" />
498     </complexType>
499
500 <complexType name="InboundOperation">
501     <sequence>
502         <any namespace="##other" processContents="lax" minOccurs="0"
503             maxOccurs="unbounded" />
504     </sequence>
505     <attribute name="name" type="NMTOKEN" use="required" />
506     <attribute name="nativeOperation" type="string" use="required" />
507     <anyAttribute namespace="##any" processContents="lax" />
508 </complexType>
509
510 <complexType name="ConnectionSpec">
511     <sequence>
512         <element name="property" type="sca:Property" minOccurs="0"
513             maxOccurs="unbounded" />
514         <any namespace="##other" processContents="lax" minOccurs="0"
515             maxOccurs="unbounded" />
516     </sequence>
517     <attribute name="type" type="NMTOKEN" use="required" />
518     <anyAttribute namespace="##any" processContents="lax" />
519 </complexType>
520
521 <complexType name="InteractionSpec">
522     <sequence>
523         <element name="property" type="sca:Property" minOccurs="0"
524             maxOccurs="unbounded" />
525         <any namespace="##other" processContents="lax" minOccurs="0"
526             maxOccurs="unbounded" />
527     </sequence>
528     <attribute name="type" type="NMTOKEN" use="required" />
529     <anyAttribute namespace="##any" processContents="lax" />
530 </complexType>
531

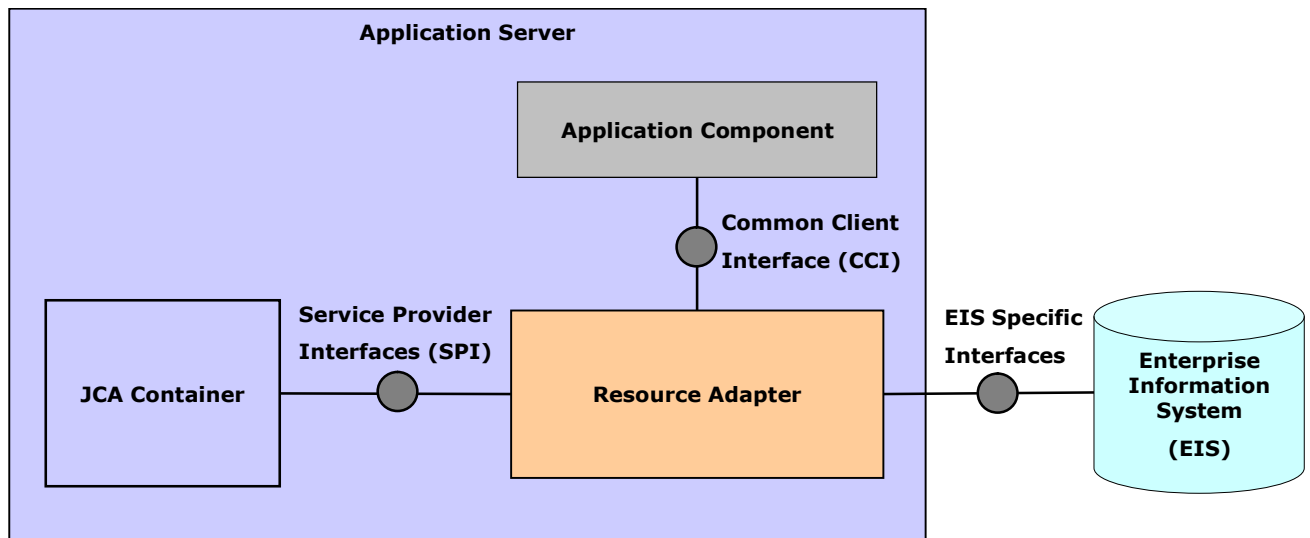
```

```
532     <element name="binding.jca" type="sca:JCABinding"  
533         substitutionGroup="sca:binding" />  
534 </schema>  
535
```

Appendix 1

JavaEE Connector Architecture Introduction

The connector architecture specification defines set of contracts that allow interoperability of the resource adapters and application server environments. The specification also defines set of client interfaces that can be optionally supported by the adapter and allow the use of adapter functionality by the application clients. The following figure illustrates the relationships of these interfaces.



The SPI defines the following management contracts that give adapter consistent view of the infrastructure provided by the server and give sever consistent view of all the adapters thus helping with integration of adapters and servers.

- Lifecycle management allows application server to control the startup of the adapter and notification to allow it to shutdown in an orderly fashion
- Work management allows the adapter to use the server resources such as threads in an efficient way and allows server to manage system resources appropriately.
- Connection management lets the server control the pooling, reusing and caching of the physical connections to the EIS system thus allowing for better scalability.
- Transactions allow the server to control EIS resource managers and provide application clients with the transactional access to external resources.
- Security contract allow for secure access to the EIS systems with security information configured and provided by the application server
- Message inflow contract allows Resource Adapter to deliver events initiated by the EIS system to the application component executing on the application server.
- Transaction inflow contract allow the application server to participate and execute in the context of the transaction initiated by the EIS system.

The CCI defines set of interfaces to access EIS functionality, through the resource adapter, from the application client. The CCI also provides access to some of the SPIs for transactions and

564 security management to allow for executions of clients running in the non-managed mode,
 565 without the presence of the Application Server.

566 **Selected JCA CCI Interfaces**

567 **Record**

```
568     public interface javax.resource.cci.Record
569         extends java.lang.Cloneable, java.io.Serializable {
570
571         public String getRecordName();
572         public void setRecordName(String name);
573         public void setRecordShortDescription(String description);
574         public String getRecordShortDescription();
575         public boolean equals(Object other);
576         public int hashCode();
577         public Object clone() throws CloneNotSupportedException;
578     }
```

579

580 **Interaction**

```
581     public interface javax.resource.cci.Interaction {
582
583         public Connection getConnection();
584         public void close() throws ResourceException;
585         public boolean execute(InteractionSpec ispec,
586             Record input, Record output) throws ResourceException;
587         public Record execute(InteractionSpec ispec,
588             Record input) throws ResourceException;
589
590     }
```

591 **MessageListener**

```
592     interface javax.resource.cci.MessageListener {
593
594         Record onMessage(Record inputData) throws ResourceException;
595     }
```

596

597 **Appendix 2 - References**

598

599

[1] SCA Specifications

600

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

601

602

[2] J2EE Connector Architecture Specification

603

<http://java.sun.com/j2ee/connector/index.jsp>