
Service Component Architecture WS-BPEL C&I Extensions for Event Processing and Pub/Sub

27 July 2009

Table of Contents

1	Introduction	3
1.1	Normative References	3
2	SCA Event Processing Extensions to WS-BPEL	4
2.1	Event Link	4
2.1.1	Consumer	4
2.1.2	Producer	4
2.2	WS-BPEL Extension Activities.....	5
2.2.1	eventConsume Extension Activity	5
2.2.2	eventProduce Extension Activity	6
3	Producers and Consumers in the Introspected Component Type	7
4	Examples.....	8
A.	XML Schema	10

1 Introduction

2 This document describes the event processing and pub/sub extensions to the SCA WS-BPEL Client and
3 Implementation Specification **[SCA-BPEL]**. It specifies how an SCA Extended WS-BPEL runtime may
4 support event processing and pub/sub described by SCA Assembly Model Specification Extensions for
5 Event Processing and Pub/Sub **[SCA-Eventing]**.

6 1.1 Normative References

- 7 **[SCA-BPEL]** OASIS Committee Draft 02, "Service Component Architecture WS-BPEL Client
8 and Implementation Specification Version 1.1," March 2009. [http://docs.oasis-](http://docs.oasis-open.org/opencsa/sca-bpel/sca-bpel-1.1-spec-cd02.pdf)
9 [open.org/opencsa/sca-bpel/sca-bpel-1.1-spec-cd02.pdf](http://docs.oasis-open.org/opencsa/sca-bpel/sca-bpel-1.1-spec-cd02.pdf)
- 10 **[SCA-Eventing]** SCA Assembly Model Specification Extensions for Event Processing and
11 Pub/Sub, April 2009.
12 http://osoa.org/download/attachments/35/SCA_Assembly_Extensions_for_Event
13 [_Processing_and_PubSub_V1_0.pdf?version=1](http://osoa.org/download/attachments/35/SCA_Assembly_Extensions_for_Event)
- 14 **[SCA-POLICY]** OASIS Committee Draft 02, "SCA Policy Framework Version 1.1," February
15 2009. [http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-](http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd02.pdf)
16 [cd02.pdf](http://docs.oasis-open.org/opencsa/sca-policy/sca-policy-1.1-spec-cd02.pdf)
- 17 **[WS-BPEL]** OASIS Standard, "Web Services Business Process Execution Language Version
18 2.0," <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf>

2 SCA Event Processing Extensions to WS-BPEL

2.1 Event Link

SCA extensions for event processing and pub/sub use the concept of events that are produced by producers and consumed by consumer through an intermediary called a channel. There is no interface contract associated either with a producer, or a consumer or a channel. Events instead may be typed using event types defined in an Event Description Language (EDL). Because of the decoupling between the producers and consumers, and the lack of interface contract, a BPEL `partnerLink`, which is typed using `partnerLinkType` (and therefore a WSDL `portType(s)`), is not suitable for event processing.

This specification introduces a new concept of event link. An event link is a way to group event activities and allow a WS-BPEL process to specify producers and consumers of events. The syntax for an `eventLink` is as follows:

```
<sca-bpel:eventLink name="xs:NCName">
  <sca-bpel:consumer requires="list of xs:QName"?>
    <sca:filters/?>
  </sca-bpel:consumer?>
  <sca-bpel:producer requires="list of xs:QName"?
    typeName="list of xs:Name"?
    typeNamespaces="list of xs:anyURI"?>?
</sca-bpel:eventLink>
```

An `eventLink` has the following attributes:

- `name` (1..1) – name of the event link. Each `eventLink` is named, and this name is used for all event interactions via that event link. Similar to a partner link name, the name of the event link must be unique within its scope.

An `eventLink` can be declared to be a consumer, producer or both, using the children elements `consumer` and/or `producer`. An `eventLink` element must contain either the `consumer` element, `producer` element or both.

An `eventLink` element can be declared anywhere a `partnerLink` can.

2.1.1 Consumer

This element declares its parent `eventLink` as a consumer of events. It has the following attributes:

- `requires` (0..n) – a list of policy intents. See SCA Policy Framework for a description of this attribute **[SCA-POLICY]**.

This element has the following children elements:

- `filters` (0..1) – filter elements. See Section 1.13 of the SCA Assembly Model Specification Extensions for Event Processing and Pub/Sub **[SCA-Eventing]** for a detailed description of filters.

2.1.2 Producer

This element declares its parent `eventLink` as a producer of events. It has the following attributes:

- `requires` (0..n) – a list of policy intents. See SCA Policy Framework for a description of this attribute **[SCA-POLICY]**.

- 60 • `typeNames` (0..1) – a list of one or more Event Type QNames for events that may be generated
61 by this producer. See the SCA Assembly Model Specification Extensions for Event Processing
62 and Pub/Sub **[SCA-Eventing]** for a detailed description of Event Types.
- 63 • `typeNamespaces` (0..1) – a list of one or more Event Type namespace URIs for events that may
64 be generated by this producer. See the SCA Assembly Model Specification Extensions for Event
65 Processing and Pub/Sub **[SCA-Eventing]** for a detailed description of Event Type namespaces.

66 2.2 WS-BPEL Extension Activities

67 This specification creates two new WS-BPEL extension activities: `eventConsume` and `eventProduce`.
68 These new extension activities are meant for event processing.

69 An event instance can contain the event data as well as metadata, (for example, creation time)
70 associated with the event instance. Similar to WS-BPEL (as it applies to message metadata access), this
71 specification does not specify how a runtime provides access to the event metadata, except for the event
72 type QName.

73 2.2.1 `eventConsume` Extension Activity

74 The `eventConsume` activity is used to consume events from the channel(s) that the consumer
75 corresponding to the event link is connected to, in the composite configuration. The event instance data
76 and type is received in the variables pointed to by the `eventConsume` element.

77 The syntax for the `eventConsume` activity is as follows:

```
78 <sca-bpel:eventConsume eventLink="xs:QName"
79                       variable="xs:NCName"?
80                       typeVariable="xs:NCName"?
81                       createInstance="yes|no"?>
82   <sca-bpel:correlations>
83     <bpel:correlation set="xs:NCName" initiate="yes|join|no" />*
84   </sca-bpel:correlations?>
85 </sca-bpel:eventConsume>
```

86 An `eventConsume` activity has the following attributes:

- 87 • `eventLink` (1..1) – event link consumer used as the event sink.
- 88 • `variable` (0..1) – WS-BPEL variable name. If specified, it is used to receive the event instance
89 data. The variable used must not be a variable that is declared using the `@messageType`
90 attribute. The XML Schema type or element used on the variable declaration must match that of
91 the event instance.
- 92 • `typeVariable` (0..1) – WS-BPEL event type variable name. If specified, it is used to specify and
93 receive the event instance type name. The variable specified here must be a variable that is
94 declared using the `@type` attribute with a value of "xs:QName". This variable is an in/out variable.
95 I.e., if this variable contains a value, then only an event instance that has the same event type
96 value as the one specified in the variable, is delivered. If the variable does not have a value, then
97 any event available for the corresponding `eventLink` consumer is delivered. On receipt, this
98 variable always contains the event instance's event type value, if present in the event instance.
- 99 • `createInstance` (0..1) – as specified in the Web Services Business Process Execution
100 Language Version 2.0 **[WS-BPEL]**.

101 An `eventConsume` activity has the following children elements:

- 102 • `correlations` (0..n) – WS-BPEL correlations element. This is similar to the `correlations`
103 element specified for the BPEL receive activity.

104 Note that WS-BPEL “standard attributes” and “standard elements” are allowed in the `eventConsume`
105 activity and are not included above for brevity.

106 WS-BPEL allows for concurrent activities. If multiple concurrent `eventConsume` activities are present, an
107 event that arrives on an event link is delivered to at most one activity within a process instance. Which
108 particular concurrent activity is picked is out of scope for this specification.

109 WS-BPEL pick activity waits for the occurrence of exactly one event from a set of events. It is possible to
110 use `eventConsume` in a pick activity along with `onMessage` and `onAlarm`.

111 WS-BPEL defines event handlers that run concurrently and get invoked when specified events occur. It is
112 possible to use `eventConsume` in `eventHandlers` along with `onEvent` and `onAlarm`.

113 2.2.2 eventProduce Extension Activity

114 The `eventProduce` activity is used to produce events for the channel(s) that the producer corresponding
115 to the event link is connected to, in the composite configuration. The event instance data and type are
116 provided by the variables pointed to by the `eventProduce` element.

117 The syntax for the `eventProduce` activity is as follows:

```
118 <sca-bpel:eventProduce eventLink="xs:QName"  
119                       variable="xs:NCName"?  
120                       typeVariable="xs:NCName">  
121   <sca-bpel:correlations>  
122     <bpel:correlation set="xs:NCName" initiate="yes|join|no" /*>  
123   </sca-bpel:correlations?>  
124 </sca-bpel:eventProduce>
```

125 An `eventProduce` activity has the following attributes:

- 126 • `eventLink` (1..1) – event link producer used as the event source.
- 127 • `variable` (0..1) – WS-BPEL variable name. If specified, it holds the event instance data. The
128 variable used must not be a variable that is declared using the `@messageType` attribute.
- 129 • `typeVariable` (0..1) – WS-BPEL event type variable name. If specified, it holds the event
130 instance type name. The variable specified here must be a variable that is declared using the
131 `@type` attribute with a value of "xs:QName".

132 An `eventProduce` activity has the following children elements:

- 133 • `correlations` (0..n) – WS-BPEL correlations element. This is similar to the `correlations`
134 element specified for the BPEL reply activity.

135 Note that WS-BPEL “standard attributes” and “standard elements” are allowed in the `eventProduce`
136 activity and are not included above for brevity.

3 Producers and Consumers in the Introspected Component Type

For each event link in the process:

1. If there is a `sca-bpel:consumer` child element then it results in an `sca:consumer` element in the introspected component type.
 - a. The value of the `@name` attribute of the `sca:consumer` element is the same as that of the `sca-bpel:eventLink` parent element, unless there is a name clash across scopes. If there is a name clash, then the algorithm defined in Section 2.3 of SCA WS-BPEL Client and Implementation Specification **[SCA-BPEL]** for name-mangling, is used to generate a unique consumer name.
 - b. The value of the `@requires` attribute of the `sca:consumer` element is the same as that of the `sca-bpel:consumer` element.
 - c. The `policySet` attribute is omitted.
 - d. The `sca:bindings` child element is omitted.
 - e. The `sca:filter` element is copied from the `sca-bpel:consumer` element to the `sca:consumer` element.
2. If there is a `sca-bpel:producer` child element then it results in an `sca:producer` element in the introspected component type.
 - a. The value of the `@name` attribute of the `sca:producer` element is the same as that of the `sca-bpel:eventLink` parent element, unless there is a name clash across scopes. If there is a name clash, then the algorithm defined in Section 2.3 of SCA WS-BPEL Client and Implementation Specification **[SCA-BPEL]** for name-mangling, is used to generate a unique producer name.
 - b. The value of the `requires`, `typeNames`, and `typeNamespaces` attributes of the `sca:producer` element are the same as that of the `sca-bpel:producer` element.
 - c. The `policySet` attribute is omitted.

163 4 Examples

164 The example below shows a process containing an event link and an `eventProduce` activity.

```
165 <process name="PublishEventProcess" ...>
166
167   ...
168
169   <!-- Event link declaration -->
170   <sca-bpel:eventLink name="POProducer">
171     <sca-bpel:producer typeNames="ns1:POEvent" />
172   </sca-bpel:eventLink>
173
174   <variables>
175     <variable name="order" element="ns1:PO"/>
176     <variable name="eventType" type="xs:QName"/>
177     ...
178   </variables>
179
180   <sequence name="main">
181     ...
182
183     <assign name="AssignPO">
184       <copy>
185         <from variable="..." />
186         <to variable="order" />
187       </copy>
188       <copy>
189         <from><literal>ns1:POEvent</literal></from>
190         <to variable="eventType" />
191       </copy>
192     </assign>
193
194     <!-- Event produce activity -->
195     <extensionActivity>
196       <sca-bpel:eventProduce eventLink="POProducer" variable="order"
197         typeVariable="eventType"/>
198     </extensionActivity>
199
200     ...
201
202   </sequence>
203
204 </process>
205
```

206 The component type introspected from the above BPEL process is the following:

```
207 <componentType ...>
208
209   <producer name="POProducer" typeNames="ns1:POEvent" />
210
211 </componentType>
```

212 The next example below shows a process containing an event link and an `eventConsume` activity.

```
213 <process name="ConsumeEventProcess" ...>
214
215   ...
216
217   <!-- Event link declaration -->
218   <sca-bpel:eventLink name="POConsumer">
```



```
219     <sca-bpel:consumer>
220         <sca:filters>
221             <sca:type qnames="ns1:POEvent" />
222         </sca:filters>
223     </sca-bpel:consumer>
224 </sca-bpel:eventLink>
225
226
227 <variables>
228     <variable name="order" element="ns1:PO"/>
229     <variable name="eventType" type="xs:QName"/>
230     ...
231 </variables>
232
233 <sequence name="main">
234     ...
235
236     <!-- Event consume activity -->
237     <extensionActivity>
238         <sca-bpel:eventConsume eventLink="POConsumer" variable="order"
239                                 typeVariable="eventType" />
240     </extensionActivity>
241     ...
242
243 </sequence>
244
245 </process>
246
247
```

248 The component type introspected from the above BPEL process is the following:

```
249 <componentType ...>
250
251     <consumer name="POConsumer">
252         <filters>
253             <type qnames="ns1:POEvent" />
254         </filters>
255     </consumer>
256
257 </componentType>
258
```

259

```

261 <schema
262   targetNamespace="http://docs.oasis-open.org/ns/opencsa/sca-bpel/200801"
263   xmlns:sca-bpel="http://docs.oasis-open.org/ns/opencsa/sca-bpel/200801"
264   xmlns:sca="http://docs.oasis-open.org/ns/opencsa/sca/200903"
265   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
266   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
267   xmlns="http://www.w3.org/2001/XMLSchema"
268   elementFormDefault="qualified">
269
270   <!-- SCA-Assembly XML Schema -->
271   <import
272     namespace="http://docs.oasis-open.org/ns/opencsa/sca/200903"
273     schemaLocation="http://docs.oasis-open.org/opencsa/sca-assembly/sca-
274 core-1.1-cd03.xsd" />
275
276   <!-- WS-BPEL 2.0 XML Schema for abstract BPEL -->
277   <import
278     namespace="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
279     schemaLocation="http://docs.oasis-
280 open.org/wsbpel/2.0/OS/process/abstract/ws-bpel_abstract_common_base.xsd" />
281
282   <!--
283     SCA BPEL eventLink. This is analogous to BPEL partnerLink for the
284     eventing world.
285   -->
286   <element name="eventLink" type="sca-bpel:EventLinkType" />
287
288   <complexType name="EventLinkType">
289     <sequence>
290       <element ref="sca-bpel:consumer" minOccurs="0" maxOccurs="1" />
291       <element ref="sca-bpel:producer" minOccurs="0" maxOccurs="1" />
292       <any namespace="##other" processContents="lax"
293         minOccurs="0" maxOccurs="unbounded" />
294     </sequence>
295     <attribute name="name" type="xsd:NCName" use="required" />
296     <anyAttribute namespace="##other" processContents="lax" />
297   </complexType>
298
299   <!--
300     SCA BPEL consumer. This can be present as a child of eventLink.
301     This results in a SCA consumer showing up in the Component Type.
302   -->
303   <element name="consumer" type="sca-bpel:ConsumerType" />
304
305   <complexType name="ConsumerType">
306     <sequence>
307       <element ref="sca:filters" minOccurs="0" maxOccurs="1" />
308       <any namespace="##other" processContents="lax"
309         minOccurs="0" maxOccurs="unbounded" />
310     </sequence>
311     <anyAttribute namespace="##other" processContents="lax" />
312   </complexType>
313
314   <!--
315     SCA BPEL producer. This can be present as a child of eventLink.
316     This results in a SCA producer showing up in the Component Type.
317   -->
318   <element name="producer" type="sca-bpel:ProducerType" />
319

```

```

320 <complexType name="ProducerType">
321   <sequence>
322     <any namespace="##other" processContents="lax"
323       minOccurs="0" maxOccurs="unbounded" />
324   </sequence>
325   <attribute name="requires" type="sca:listOfQNames" />
326   <attribute name="typeNames" type="sca:listOfQNames" />
327   <attribute name="typeNamespaces" type="sca:listOfAnyURIs" />
328   <anyAttribute namespace="##other" processContents="lax" />
329 </complexType>
330
331 <!--
332   SCA BPEL consume extension activity. This activity allows one to consume
333   an event.
334 -->
335
336 <element name="eventConsume" type="sca-bpel:EventConsume" />
337
338 <complexType name="EventConsume">
339   <complexContent>
340     <extension base="bpel:tActivity">
341       <attribute name="eventLink" type="QName" use="required" />
342       <attribute name="variable" type="bpel:BPELVariableName" />
343       <attribute name="typeVariable" type="bpel:BPELVariableName" />
344       <attribute name="createInstance" type="bpel:tBoolean"
345         default="no" />
346     </extension>
347   </complexContent>
348 </complexType>
349
350 <!--
351   SCA BPEL produce extension activity. This activity allows one to produce
352   an event.
353 -->
354 <element name="eventProduce" type="sca-bpel:EventProduce" />
355
356 <complexType name="EventProduce">
357   <complexContent>
358     <extension base="bpel:tActivity">
359       <attribute name="eventLink" type="QName" use="required" />
360       <attribute name="variable" type="bpel:BPELVariableName" />
361       <attribute name="typeVariable" type="bpel:BPELVariableName" />
362     </extension>
363   </complexContent>
364 </complexType>
365
366 <element name="correlations" type="bpel:tCorrelations" />
367
368 </schema>

```