



---

# WSRF Application Notes

## Committee Draft 02, 25 March 2006

**Document identifier:** wsr-application\_notes-1.2-notes-cd-02

**Location:**

[http://docs.oasis-open.org/wsr/wsr-application\\_notes-1.2-cd-02.pdf](http://docs.oasis-open.org/wsr/wsr-application_notes-1.2-cd-02.pdf)

**Editors:**

Katy Warr <[katy\\_warr@uk.ibm.com](mailto:katy_warr@uk.ibm.com)>

Roger Menday <[r.menday@fz-juelich.de](mailto:r.menday@fz-juelich.de)>

**Abstract:**

This document describes how applications might use the Web Services Resource Framework (WSRF) family of specifications.

**Status:**

This document is published by this TC as a "committee draft".

Committee members should send comments on this specification to the [wsrf@lists.oasis-open.org](mailto:wsrf@lists.oasis-open.org) list. Others may submit comments to the TC via the web form found on the TC's web page at <http://www.oasis-open.org/committees/wsr>. Click the button for "Send A Comment" at the top of the page. Submitted comments (for this work as well as other works of that TC) are publicly archived and can be viewed at <http://lists.oasis-open.org/archives/wsr-comment/>.

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 WSRF TC web page (<http://www.oasis-open.org/committees/wsr/>).

26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63

## Table of Contents

Table of Contents .....	2
1 Introduction .....	3
2 Who Should Read This Document? .....	4
3 AppNotes Relating to Resource Properties .....	5
3.1 Defining a Derived WS-Resource in WSDL 1.1 .....	5
3.1.1 Best Practice and Examples .....	5
3.2 Defining a Resource Property that Must Always Exist .....	10
3.3 Invalid or Non-existent Resource Properties .....	10
3.3.1 Best Practice and Examples .....	10
3.4 Resource Property Attributes .....	11
3.4.1 Best Practice and Examples .....	11
3.5 Adding Attributes to the Root of the Resource Properties Document .....	13
3.6 Notifying Clients of Resource Property Changes .....	13
3.6.1 Best Practice and Examples .....	14
4 AppNotes Relating to Lifecycle .....	15
4.1 Resource Lifecycle Management .....	15
4.1.1 Best Practice and Examples .....	15
5 AppNotes Relating to Base Faults .....	16
5.1 Fault Handling .....	16
5.1.1 Best Practice and Examples .....	16
6 AppNotes Relating to ServiceGroup .....	19
6.1 ServiceGroup Members with Derived PortTypes .....	19
6.1.1 Best Practices and Examples .....	19
7 AppNotes Relating to Resource Metadata .....	20
7.1 Derived Resource Metadata .....	20
7.1.1 Best Practices and Examples .....	20
8 Interoperability Considerations .....	21
8.1 Interoperability .....	21
8.2 Intermediary Changes of Namespace Prefixes .....	21
8.2.1 Best Practices and Examples .....	21
8.3 Normative .....	22
8.4 Non-Normative .....	22
Appendix A. Acknowledgments .....	23
Appendix B. Revision History .....	24
Appendix C. Notices .....	25

---

64

## 1 Introduction

65 This document presents non-normative information that may be of benefit to WS-Resource  
66 application developers.

67 The purpose of this document is to answer common questions that might arise during WSRF  
68 application development by means of non-normative scenarios and examples. Additionally,  
69 this document may also serve as an aid to clarify potential usage scenarios of the Web  
70 Services Resource Framework (WSRF). The intended audience of this document is WS-  
71 Resource application designers

72 This document is divided into sections, each addressing a different aspect of WS-Resource  
73 application development. Each section contains information on best practices, along with  
74 examples, or references to examples elsewhere.

---

75

## 2 Who Should Read This Document?

76 This Application Notes (AppNotes) document provides a guide for WSRF application  
77 developers who already understand the fundamentals of developing a WSRF application. The  
78 AppNotes is likely to be used as a reference document in order to answer “frequently asked  
79 questions” that might arise during application development. For example: How should multiple  
80 WSRF portTypes be combined to produce a single derived portType? Where there is no  
81 definitive answer to a question, the AppNotes provides a recommended best practice. This  
82 document assumes familiarity with the WSRF specifications and the examples introduced in  
83 the **[WSRFPrimer]**.

84 This document does not present a full end-to-end WSRF example and is not meant as an  
85 entry point to WSRF. For a WSRF tutorial, readers should refer to the **[WSRFPrimer]**.

86 This document deals first with implementation considerations relating to resource properties  
87 **[WS-ResourceProperties]** since the presence of a resource properties document is a  
88 defining characteristic of a WS-Resource. Implementation considerations for the remaining  
89 WSRF specifications follow. For normative descriptions of WSRF, readers should refer to the  
90 WSRF set of specifications: **[WS-Resource]**, **[WS-ResourceProperties]**, **[WS-**  
91 **ResourceLifetime]**, **[WS-BaseFaults]** and **[WS-ServiceGroup]**.

---

## 92 3 AppNotes Relating to Resource Properties

### 93 3.1 Defining a Derived WS-Resource in WSDL 1.1

94 A designer of a WS-Resource application may need to derive a WS-Resource portType by  
95 extending or aggregating one or more existing WS-Resource portType(s). The newly created  
96 portType will also have an associated resource properties document that may be derived from  
97 the existing portType(s). There is a trade-off in portType design between the freedom of the  
98 designer to design a portType as he/she wishes and the ability to extend the portType in  
99 future (which may place restrictions on the way that a resource properties document is  
100 assembled).

#### 101 3.1.1 Best Practice and Examples

##### 102 3.1.1.1 Resource Properties and Interface Aggregation

103 Web service interface designers may define a collection of discrete interfaces (portTypes),  
104 each of which defines a set of message exchange patterns (operations). A common design  
105 scenario is one in which the designer combines these discrete interfaces to form a composed,  
106 most-derived interface of a Web service. Examples of independently-specified interfaces  
107 designed for purposes of aggregation into a most-derived interface include portTypes defined  
108 by WS-ResourceProperties [**WS-ResourceProperties**], WS-BaseNotification [**WS-**  
109 **BaseNotification**], WS-ResourceLifetime [**WS-ResourceLifetime**], and a large number of  
110 general-purpose or application-domain-specific management interfaces. Further, there may  
111 be various dependencies between these interfaces. For example, the NotificationProducer  
112 portType publishes a list of subscribable topics as a resource property, accessed using the  
113 operations from the WS-ResourceProperties portType.

114 Within WSDL 1.1, there is no formally-defined interface extension mechanism<sup>1</sup>. In WSDL 1.1  
115 we expect service designers to copy-and-paste operations from the various constituent  
116 interfaces into a single, flat, most-derived service interface. In addition, we expect the service  
117 interface designer to compose a resource properties document for the most-derived Web  
118 service interface that consists of all resource property element declarations from each of the  
119 constituent interfaces used in the composition.

120 In the following example, a designer wishes to extend the Printer portType described in  
121 [**WSRFPrimer**]. The particular portType enables clients to print and create print jobs. An  
122 extract from the Printer schema and WSDL follows:

---

<sup>1</sup> WSDL 2.0 is expected to define a mechanism to formally model interface aggregation  
/interface/@extends [WSDL 2.0].

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

```
<xsd:schema
  xmlns:pr="http://docs.oasis-open.org/wsrf/Printer.xsd"
  targetNamespace="http://docs.oasis-open.org/wsrf/Printer.xsd"
  ...>
  ...
  <xsd:element name="PrinterRP">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="pr:printer_reference" />
        <xsd:element ref="pr:printer_name" />
        <xsd:element ref="pr:printer_state" />
        <xsd:element ref="pr:printer_is_accepting_jobs" />
        <xsd:element ref="pr:queued_job_count" />
        <xsd:element ref="pr:document_format_supported" />
        <xsd:element ref="pr:job_hold_until_default"
          minOccurs="0" />
        <xsd:element ref="pr:job_hold_until_supported"
          minOccurs="0" maxOccurs="unbounded" />
        <xsd:element ref="wsrf-rp:QueryExpressionDialect"
          minOccurs="0" maxOccurs="unbounded" />
        <xsd:element ref="pr:job_properties"
          minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

```
<wsdl:definitions ...
  xmlns:prw=http://docs.oasis-open.org/wsrf/Printer.wsdl
  xmlns:pr=http://docs.oasis-open.org/wsrf/Printer.xsd
  ...>
  <wsdl:types>
    <xsd:schema
      targetNamespace="http://docs.oasis-open.org/wsrf/Printer.xsd" ... >
      <xsd:import
        namespace="http://docs.oasis-open.org/wsrf/Printer.xsd" ... />
      </xsd:schema>
    </wsdl:types>
    ...
    <!-- Association of resource properties document to a portType -->
    <wsdl:portType name="Printer"
      wsrf-rp:ResourceProperties="pr:PrinterRP">

      <!--Operations supported by the Print portType -->
      <wsdl:operation name="Print_Job"> ... </wsdl:operation>
      <wsdl:operation name="Create_Job"> ... </wsdl:operation>

      <!-- WSRF operations supported by this portType -->
      <wsdl:operation
        name="GetResourcePropertyDocument">...</wsdl:operation>
      <wsdl:operation name="GetResourceProperty">...</wsdl:operation>
      <wsdl:operation
        name="GetMultipleResourceProperties">...</wsdl:operation>
      <wsdl:operation
        name="QueryResourceProperties">...</wsdl:operation>
      <wsdl:operation
        name="SetResourceProperties">...</wsdl:operation>
    </wsdl:portType>
```

184  
185

```
</wsdl:definitions>
```

186

187 The designer wishes to extend the Printer<sup>2</sup> portType to cater for documents defined by URI.  
188 Thus, we require support for the following two new operations:

- 189 • Print\_URI – Submit a document for printing – document identified by the URI.
- 190 • Send\_URI – Add to a multi-document job – document identified by the URI.

191 The new portType will be called URI\_Printer. It extends the more generic Printer portType.

192 The procedure for the derivation of this new portType is an example of manual interface  
193 aggregation in WSDL 1.1, using copy-and-paste. A recommendation for the derivation is as  
194 follows:

- 195 • Define the new portType.

196       In this example the new portType is named “URI\_Printer”. This portType extends  
197       “Printer”.

- 198 • Copy all of the operation child elements from the portType being extended, and paste  
199       them as child elements of the new portType; the order of the operations should be  
200       preserved.

201       In this example, the “Print\_Job” and “Create\_Job” operations are copied from the  
202       Printer portType and pasted as child elements of the URI\_Printer portType.

- 203 • Define new additional operations as child elements of the new portType. When one is  
204       extending an already-defined portType, this simply requires a copy-and-paste into the  
205       new portType.

206       In this example, the “Print\_URI” and “Send\_URI” operations are new operations  
207       defined by the URI\_Printer portType.

- 208 • Define a new resource properties document, as an XML global element declaration,  
209       following the requirements defined in **[WS-ResourceProperties]**.

210       In this example, the element is named “uri\_printer\_property” and defined in a new  
211       “http://docs.oasis-open.org/wsrf/URIPrinter.xsd” namespace that imports and extends  
212       the “http://docs.oasis-open.org/wsrf/Printer.xsd” namespace.

- 213 • Copy all of the resource property elements from the resource properties document of the  
214       portType being extended, and paste them as child elements of the new resource  
215       properties document; the order of the elements should be preserved. This step must be  
216       repeated for each portType that is being extended by this new portType. Any duplicate  
217       child elements must be removed.

---

<sup>2</sup> Note that the Printer portType has already been aggregated with operations and resource properties from WS-ResourceProperties.

218 In this example, the resource property elements such as "pr:printer\_name" or  
219 "pr:printer\_state" are copied from the Printer resource properties document  
220 declaration and pasted to the URI\_Printer resource properties document declaration.

- 221 • Define any additional resource property elements that are specific to the newly-defined  
222 resource properties document type.

223 The resulting URI\_Printer schema and WSDL is as follows:

224

```
225 <xsd:schema  
226 xmlns:pr="http://docs.oasis-open.org/wsrf/Printer.xsd"  
227 xmlns:expr="http://docs.oasis-open.org/wsrf/URIPrinter.xsd"  
228 targetNamespace="http://docs.oasis-open.org/wsrf/URIPrinter.xsd"  
229 ...>  
230 <xsd:import  
231 namespace=http://docs.oasis-open.org/wsrf/Printer.xsd  
232 ... />  
233 <xsd:element name="extended_printer_properties">  
234 <xsd:complexType>  
235 <xsd:sequence>  
236 <xsd:element ref="pr:printer_reference" />  
237 <xsd:element ref="pr:printer_name" />  
238 <xsd:element ref="pr:printer_state" />  
239 <xsd:element ref="pr:printer_is_accepting_jobs" />  
240 <xsd:element ref="pr:queued_job_count" />  
241 <xsd:element ref="pr:document_format_supported" />  
242 <xsd:element ref="pr:job_hold_until_default"  
243 minOccurs="0" />  
244 <xsd:element ref="pr:job_hold_until_supported"  
245 minOccurs="0" maxOccurs="unbounded" />  
246 <xsd:element ref="wsrf-rp:QueryExpressionDialect"  
247 minOccurs="0" maxOccurs="unbounded" />  
248 <xsd:element ref="pr:job_properties"  
249 minOccurs="0" maxOccurs="unbounded" />  
250 <xsd:element ref="expr:uri_printer_property"/>  
251 </xsd:sequence>  
252 </xsd:complexType>  
253 </xsd:element>  
254 </xsd:schema>
```

255

```
256 <wsdl:definitions ...  
257 xmlns:pr="http://docs.oasis-open.org/wsrf/Printer.wsdl"  
258 xmlns:expr="http://docs.oasis-open.org/wsrf/URIPrinter.xsd"  
259 xmlns:exprw="http://docs.oasis-open.org/wsrf/URIPrinter.wsdl"  
260 ...>  
261 ...  
262 <wsdl:types>  
263 <xsd:schema targetNamespace="http://docs.oasis-  
264 open.org/wsrf/URIPrinter.xsd" ... >  
265 <xsd:import  
266 namespace="http://docs.oasis-open.org/wsrf/URIPrinter.xsd" ... />  
267 </xsd:schema>  
268 </wsdl:types>  
269 ...  
270 <!-- Association of resource properties document to a portType -->  
271 <wsdl:portType name="URI_Printer"  
272 wsrf-rp:ResourceProperties=  
273 "expr:extended_printer_properties">  
274 ...  
275 <!--Operations supported by the Printer portType -->  
276 <wsdl:operation name="Print_Job"> ... </wsdl:operation>
```

277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296

```
<wsdl:operation name="Create_Job"> ... </wsdl:operation>

<!-- WSRF operations supported by this portType -->
<wsdl:operation
name="GetResourcePropertyDocument">...</wsdl:operation>
<wsdl:operation name="GetResourceProperty">...</wsdl:operation>
<wsdl:operation
name="GetMultipleResourceProperties">...</wsdl:operation>
<wsdl:operation
name="QueryResourceProperties">...</wsdl:operation>
<wsdl:operation
name="SetResourceProperties">...</wsdl:operation>

<!-- Operations supported by the URI_Printer portType -->
<wsdl:operation name="Print_URI"> ... </wsdl:operation>
<wsdl:operation name="Send_URI"> ... </wsdl:operation>

</wsdl:portType>
</wsdl:definitions>
```

### 297 **3.1.1.2 Creating a New Port Type by Aggregating Existing PortTypes**

298 In the section above we showed how it is possible to create a new portType by adding  
299 operations and resource properties into the definition of a most-derived portType. Sometimes  
300 a WS-Resource service designer may need to create a portType based on two or more  
301 existing portTypes, where one or more of these existing portTypes may have an associated  
302 resource properties document.

303 For example, we may wish to extend the URI\_Printer portType with operations and resource  
304 properties from a previously defined AdminPortType. We may also wish to preserve the  
305 original source of the operations in the composed portType (see following section).  
306 Essentially, the process to follow is the same as above: we just use the already-existing  
307 WSDL as input to the copy-and-paste process.

### 308 **3.1.1.3 Establishing Aggregated PortType Derivation**

309 There are scenarios where a consumer of a portType that has been created by aggregating a  
310 more generic portType needs to discover this derivation in order to find, and act upon, the  
311 WS-Resource that is exposed by the portType.

312 A recommended pre-emptive solution to this problem is to always include sufficient  
313 documentation in WS-Resource design so that the origins of each operation or resource  
314 property can easily be established.

315 In addition, we can use a mechanism defined by the WS-Addressing WSDL binding to help  
316 identify which portType an operation is derived from. Following the recommendation already  
317 described, copy all the operation child elements from the portType being extended, and paste  
318 them as child elements of the new portType.

319 Specify, in the WSDL, a wsa:Action attribute for each wsdl:input and wsdl:output message.  
320 Specifically:

- 321 • If the wsa:Action attribute does not appear in the child input or output element, then  
322 define the wsa:Action for the child input or output element as:  
323 [target namespace]/[port type name]/[input/output name].
- 324 • If the wsa:Action attribute does not appear in the child fault elements then define the  
325 wsa:Action for the child fault elements as:  
326 [target namespace]/[port type name]/[operation name]Fault:[fault name]

327 where:

328 [target namespace] is the /definition/@targetnameNamespace of the port type from which  
329 the operation was copied  
330 [port type name] is the /definition/porttype/@name of the portType from which the  
331 operation was copied  
332 [input/output name] is the name of the element as defined in WSDL 1.1, section 2.4.5.  
333 [fault name] is the /definition/porttype/operation/fault/@name  
334 For example:  
335 wsa:Action="http://docs.oasis-open.org/wsrf/Printer.wsdl/Printer/[input/output name]"

#### 336 **3.1.1.4 Creating a New PortType by Aggregating Existing PortTypes** 337 **which have a Resource Property with the Same Name but** 338 **Different Multiplicity**

339 There may be occasions when a WS-Resource service designer creates a portType based on  
340 two or more existing portTypes whose resource properties documents contain a property with  
341 the same QName.

342 This scenario presents no problem as long as the semantics (in particularly, the multiplicity) of  
343 the shared property are the same in both cases. However, if the semantics of this property  
344 differ between the existing portTypes, there may be ambiguity regarding the derived  
345 property's behavior. For example, the property is defined as mandatory in one of the derived-  
346 from portTypes, and optional in the other.

347 The best solution to this problem is to revisit the design of the derived-from portTypes and  
348 consider why the same property occurs in each with different multiplicity. The portTypes  
349 should be restructured to remove the multiplicity conflict.

350 If this is not possible, (i.e. the derived-from portTypes cannot be changed), a judgment will  
351 need to be made. For example, the application designer may choose to take the least  
352 restrictive of the conflicting multiplicities.

### 353 **3.2 Defining a Resource Property that Must Always Exist**

354 An attempt to delete a resource property whose minOccurs>0, using the  
355 DeleteResourceProperties message exchange pattern (MEP) (or the  
356 DeleteResourceProperties component of a SetResourceProperties MEP), will result in the  
357 InvalidModificationFault because deletion of this property would render the resource  
358 properties document invalid.

### 359 **3.3 Invalid or Non-existent Resource Properties**

360 This section describes how to distinguish between properties that are valid, invalid, or  
361 unavailable, and those that are available but have no current value assigned.

#### 362 **3.3.1 Best Practice and Examples**

##### 363 **3.3.1.1 Establishing a List of Valid Resource Properties**

364 A client wishing to establish whether a resource property is present in a resource property  
365 document should use the GetResourceProperty request on a resource, passing the property  
366 name in question. Since resource properties may be dynamically inserted into or deleted  
367 from a resource properties document, the returned list of valid resource properties may vary  
368 for a particular document over time. Additionally, if the resource properties schema allows  
369 open content via an xsd:any element, then even the list of valid resource property QNames  
370 may vary in a particular document over time.

371 A client wishing to establish all the resource properties that are present for a resource should  
372 issue the `GetResourcePropertiesDocument` method on the resource.

373 An application developer may consider using an additional resource property to publish a list  
374 of QNames of resource properties that are valid and/or present, for a resource that supports  
375 open content in its resource properties document. In many cases this may prove more  
376 economical than retrieving the entire resource properties document.

### 377 **3.3.1.2 Invalid Properties**

378 When a `GetResourceProperty` operation returns an `InvalidResourcePropertyQNameFault`, the  
379 property requested does not exist in the resource properties document schema for the WS-  
380 Resource.

381 Similarly, a client receives an `InvalidResourcePropertyQNameFault` in response to a  
382 `GetMultipleResourceProperties` request when one or more of the resource property QNames  
383 in the request message do not correspond to resource properties in the resource property  
384 document schema.

### 385 **3.3.1.3 Properties That Are Valid but Have No Value**

386 If a client issues `GetResourceProperty` on a property described in the schema with  
387 `minOccurs=0`, the `GetResourceProperty` operation returns an empty  
388 `GetResourcePropertyResponse` message if there is currently no resource property of this  
389 type in the resource properties document.

390 Similarly, the `GetMultipleResourceProperties` operation returns a collection of the properties  
391 corresponding to the QNames in the request message. In the case where a resource  
392 properties document does not contain a value for one of the requested properties, no element  
393 is added to the collection for that property's QName.

### 394 **3.3.1.4 Properties That Have the Value 'nil'**

395 If a property is declared nillable and it has the value nil, the `GetResourceProperty` operation  
396 will return the resource property element decorated with an `xsi:nil="true"` attribute.

397 Similarly, the `GetMultipleResourceProperties` operation returns a collection of the properties  
398 corresponding to the QNames in the request message. Any property whose value is nil will  
399 be decorated with the `xsi:nil="true"` attribute.

400 The `TerminationTime` resource property defined in **[WS-ResourceLifetime]** is an example of  
401 a nillable Resource Property. A service requestor can specify that there is no scheduled  
402 termination time for the WS-Resource by using the `SetResourceProperty` operation to set the  
403 value of this Resource Property to nil:

404 

```
<wsr:TerminationTime xsi:nil="true"/>.
```

## 405 **3.4 Resource Property Attributes**

406 WS-Resources may associate attributes with individual resource property definitions in order  
407 to indicate that instances of that resource property will exhibit a particular behavioral trait.  
408 This section explains how this is done and gives some examples illustrating why it might be  
409 useful.

### 410 **3.4.1 Best Practice and Examples**

#### 411 **3.4.1.1 Refining a Resource Property Definition with Lifetime Attributes**

412 Consider a WS-Resource that represents an item in a warehouse. The item may have a price  
413 associated (by means of a resource property) and, on occasion, a sale price. In this example,

414 the resource property defining the sale price is only valid between specific dates, so it is  
415 necessary to indicate to the consumer some lifetime aspects of this property. In this case  
416 GoodFrom and GoodUntil are attributes added to the sale price resource property to indicate  
417 the date that the sale price came into effect and the date until when it is valid. Attributes such  
418 as GoodFrom and GoodUntil can appear on a resource property if:

- 419 • the resource property definition explicitly includes them as attributes

420 or if:

- 421 • the resource property definition allows attribute extensibility by associating anyAttribute  
422 with the property definition (as described in the example in section 3.4.1.2, Providing  
423 Attribute Extensibility to a Resource Property).

424 The definition of the sale price resource property with GoodFrom and GoodUntil explicitly  
425 defined as attributes is illustrated by the following XML schema:

426

```
427 <xsd:element name="SalePrice">  
428   <xsd:complexType>  
429     <xsd:simpleContent>  
430       <xsd:extension base="xsd:decimal">  
431         <xsd:attribute name="GoodFrom" type="xsd:dateTime"/>  
432         <xsd:attribute name="GoodUntil" type="xsd:dateTime"/>  
433       </xsd:extension>  
434     </xsd:simpleContent>  
435   </xsd:complexType>  
436 </xsd:element>
```

437 The following is an example SalePrice resource property:

```
438 <SalePrice  
439   GoodFrom="2006-01-23T09:59:55.025+01:00"  
440   GoodUntil="2006-01-23T12:59:55.093+01:00" ...>  
441   546.33  
442 </SalePrice>
```

### 443 3.4.1.2 Providing Attribute Extensibility to a Resource Property

444 The CurrentTime resource property defined in [WS-ResourceLifetime] is an example of a  
445 Resource Property definition that allows attribute extensibility by associating anyAttribute with  
446 the property definition.

447 The XML schema definition for the CurrentTime resource property element type is as follows:

448

```
449 <xsd:element name="CurrentTime">  
450   <xsd:complexType>  
451     <xsd:simpleContent>  
452       <xsd:extension base="xsd:dateTime">  
453         <xsd:anyAttribute namespace="##other"  
454           processContents="lax"/>  
455       </xsd:extension>  
456     </xsd:simpleContent>  
457   </xsd:complexType>  
458 </xsd:element>
```

459

460 The xsd:anyAttribute in this resource property enables application designers to associate an  
461 attribute that has not been pre-specified with the CurrentTime. An example might be an  
462 attribute indicating the accuracy of the CurrentTime property.

### 3.5 Adding Attributes to the Root of the Resource Properties Document

No restrictions are placed on the adding of attributes to the resource properties document root. Consumers of the WS-Resource can retrieve attributes on the resource properties document root by exploiting the GetResourcePropertyDocument MEP.

For example, the PrinterRP resource properties document declaration introduced earlier might be extended as follows to introduce a mandatory attribute to record the creation date of the printer resource:

```
<xsd:element name="PrinterRP">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="pr:printer_reference" />
      <xsd:element ref="pr:printer_name" />
      ...
    </xsd:sequence>
    <xsd:attribute name="CreationDate" type="xsd:dateTime"
                  use="required" />
  </xsd:complexType>
</xsd:element>
```

An extract of the resource property document instance is:

```
<PrinterRP CreationDate="2006-01-23T12:25:54.507+01:00" ...>
  ...
</PrinterRP>
```

In order to query attributes on the root of the resource properties document, a WS-Resource client may extract the complete resource properties document using the GetResourcePropertyDocument MEP and query the specific attribute(s). It is also possible to use the QueryResourceProperties operation (when available) forming a query which reads the root attributes<sup>3</sup>.

Similarly, to perform a resource properties document root attribute update, the client application must update the attribute in a local copy of the resource properties document and then update the resource properties document associated with the resource, using the PutResourcePropertyDocument MEP.

### 3.6 Notifying Clients of Resource Property Changes

A common pattern for WS-Resource applications is for the WS-Resource to send automatic notifications of changes to resource property elements of its resources to interested partners.

---

<sup>3</sup> We refer the reader to section 8.2 for interoperability recommendations regarding namespace prefixes and XPath queries.

498 Resource properties may be changed by external events (for example, by use of  
499 SetResourceProperties) or by events internal to the service and not directly visible to the  
500 client (for example, a printer is put out of service).

## 501 **3.6.1 Best Practice and Examples**

### 502 **3.6.1.1 Resource Property Value-Change Notification Pattern**

503 In order to provide automatic notification of resource property changes, application designers  
504 should compose WSRF with **[WS-BaseNotification]**.

505 A normative description of how this should be done is contained in **[WS-**  
506 **ResourceProperties]**, and an example is contained in the **[WSRFPrimer]**.

### 507 **3.6.1.2 Format of the Resource Property Value-Change Notification**

508 The notification messages can take many forms. The application developer should be aware  
509 that the ResourcePropertyValueChangeNotification element may occur anywhere within the  
510 notification message. Likewise, a notification subscriber should ensure that a notification  
511 consumer is able to process the particular form of the notification message sent by the  
512 producer.

513 The element may appear as the root element of the notification message, or as a descendent  
514 of the root element, accommodating patterns where the notification message is contained in  
515 an enveloping mechanism. For example, Web Services Distributed Management **[WSDM]**  
516 defines a format for management event messages, which may then embed a  
517 ResourcePropertyValueChangeNotification element.

---

## 518 4 AppNotes Relating to Lifecycle

### 519 4.1 Resource Lifecycle Management

520 This section describes how individual resources that are fronted by a WS-Resource Web  
521 service are created and destroyed.

#### 522 4.1.1 Best Practice and Examples

##### 523 4.1.1.1 Resource Creation

524 The specifications in the WSRF family do not specify how a WS-Resource instance should be  
525 created. A commonly used pattern is the “factory pattern”, whereby a separate Web service  
526 (factory) exposes an operation to clients for creating a new resource instance and returning a  
527 reference to that instance.

528 The explicit factory pattern is by no means the only mechanism by which resource instances  
529 might be created. A resource might be created as part of an operation performing a wider  
530 function. For example, in WS-Notification, a subscription request creates an endpoint to  
531 represent the subscription and returns the relevant endpoint reference (EPR) as part of its  
532 behavior.

533 Refer to the [WSRFPrimer] for examples of resource creation, such as the creation of a print  
534 job when a document is printed.

##### 535 4.1.1.2 Resource Destruction

536 The WS-Resource Lifetime specification defines the mechanisms by which the lifecycles of  
537 resources should be managed. Resources can be explicitly destroyed or scheduled for  
538 destruction. Refer to [WS-ResourceLifetime] for details.

539 A common practice is to specify the resource's initial termination time as part of its creation –  
540 thus saving an operation.

---

541

## 5 AppNotes Relating to Base Faults

542

### 5.1 Fault Handling

543

Problem determination is an important aspect of Web service application development. This section describes how best to generate faults in the event of an error, and how a client of a Web service might process the faults in order determine the underlying problem.

544

545

This section describes best practices for fault generation and consumption in a WS-Resource application environment.

546

547

548

#### 5.1.1 Best Practice and Examples

549

A WS-Resource application should adhere to the **[WS-BaseFaults]** specification for all its fault processing. Adhering to this standard has a number of benefits:

550

551

- The **[WS-BaseFaults]** model removes the need for proprietary or application-specific fault handling. Fault recipients may therefore be developed in isolation from the service generating the fault.

552

553

554

- **[WS-BaseFaults]** provides a simple and powerful pattern for fault processing, enabling the application designer to focus on the design of the application rather than the underlying fault processing model.

555

556

557

- A standard model for fault processing enables re-use of code both in the recipient of the fault and in the fault generator. This also eases the development of tools.

558

559

Faults are generated either in response to errors in the Web service application (service faults) or as a result of some kind of system processing error (system faults). For example, a system fault may be generated as part of transport processing. This section deals with the handling of service faults, but it is worth noting that a service might raise a service fault as a result of an underlying system fault cause.

560

561

562

563

564

##### 5.1.1.1 Defining and Generating Base Faults

565

Each base fault that might be generated by a service requires its own distinct XML schema type that extends `wsrf-bf:BaseFault`. This extended fault complexType may contain additional attributes and/or elements.

566

567

568

Clearly, the type of information that should be defined in an extended fault type depends on the application and its deployment. Ensuring that the correct information is available in the fault is critical to effective problem determination. Here are some recommended additional elements that might be contained in the extended fault:

569

570

571

572

- **Host:** The host on which the fault was generated. If the service could be run on more than one host (for example, in the case of load balancing), it is important to include the host name as an element in the extended fault type.

573

574

575

- **Process:** The process in which the service was running when the fault occurred.

576

577

- **SOAPFault information:** If the `WS-BaseFault` is to wrap a SOAP fault then the SOAP fault code and role should be contained in the fault.

578

579

580

581

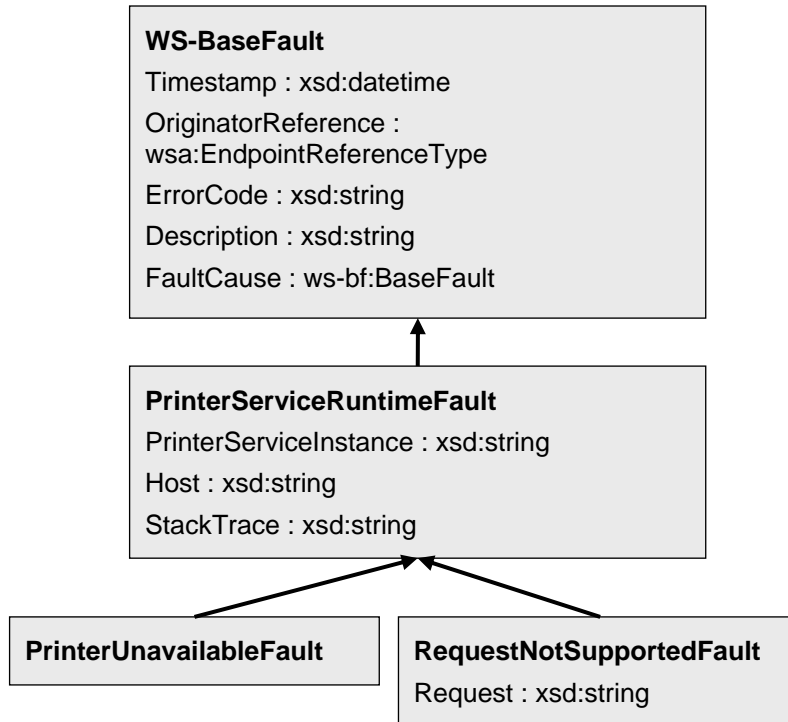
Consider the Printer example used previously. We require extended fault types for use by our printer portType. To begin, it would be prudent to define a generic extended fault for the printer service from which all other printer service faults could be derived. This fault type will define the basic information that we require in every fault generated by the printer service.

582

583

For example, every fault generated should contain an identifier specifying the printer to which the fault was directed, the host, and a stack trace detailing the fault cause. Further extended

584 faults may be generated from this underlying generic type (for example:  
 585 PrinterUnavailableFault and RequestNotSupportedFault).  
 586



587  
 588

589 An example of a PrinterServiceRuntimeFault SOAP 1.1 message follows:

```

590 <s11:Envelope
591   xmlns="http://schemas.xmlsoap.org/soap/envelope/"
592   xmlns:s11="http://schemas.xmlsoap.org/soap/envelope/"
593   xmlns:wsa=" http://www.w3.org/2005/08/addressing"
594   xmlns:wsrf-bf="http://docs.oasis-open.org/wsrf/bf-2"
595   xmlns:wsrf-printer="http://docs.oasis-open.org/wsrf/printer">
596 <s11:Header>
597   <wsa:Action>
598     http://docs.oasis-open.org/wsrf/fault
599   </wsa:Action>
600   ...
601 </s11:Header>
602 <s11:Body>
603   <s11:Fault>
604     <faultcode>s11:Client</faultcode>
605     <faultstring>No such resource exists</faultstring>
606     <faultactor>http://example.org/someactor</faultactor>
607     <detail>
608       <wsrf-printer:PrinterServiceRuntimeFault>
609         <wsrf-bf:Timestamp>
610           2005-05-04T20:18:44.970Z
611         </wsrf-bf:Timestamp>
612         <wsrf-printer:Host>
613           123.434.434.33
614         </wsrf-printer:Host>
615       </wsrf-printer:PrinterServiceRuntimeFault>
616     </detail>
617   </s11:Fault>
618 </s11:Body>
  
```

619

```
</s11:Envelope>
```

620 In the example above, the PrinterServiceInstance and Host elements are not present within  
621 the PrinterServiceRuntimeFault as they are optional.

### 622 **5.1.1.2 Relaying Base Faults**

623 Faults received by an intermediary should be wrapped by a fault containing the intermediary  
624 information and then relayed. The wrapped fault is contained in the FaultCause element of  
625 the new enclosing fault.

626

## 6 AppNotes Relating to ServiceGroup

627

### 6.1 ServiceGroup Members with Derived PortTypes

628

Often a ServiceGroup [WS-ServiceGroup] will be constructed which contains member services with derived portTypes (see section 3.1 for derivation procedure guidelines). This section clarifies the application of the ServiceGroups specification for services with derived portTypes.

629

630

631

632

#### 6.1.1 Best Practices and Examples

633

The MembershipContentRule resource property element of a ServiceGroup defines the conditions that must be met by a ServiceGroupEntry of a ServiceGroup. A ServiceGroupEntry may be restricted by the MemberInterfaces and ContentElements attributes of the MembershipContentRule.

634

635

636

637

##### 6.1.1.1 MemberInterfaces Attribute for Derived PortTypes

638

The ServiceGroup specification refers only to normatively-defined information in other specifications. Therefore, the most derived WSDL 1.1 portType is the name to be used in the MemberInterfaces attribute of the MembershipContentRule. For example, from the derived Printer portType of section 3.1.1.1, we could specify the following MembershipContentRule:

639

640

641

642

643

```
<wsrf-sg:MembershipContentRule MemberInterfaces="prw:URI_Printer" />
```

644

##### 6.1.1.2 ContentElements Attribute for Derived PortTypes

645

It may be convenient to build a ServiceGroup whose member services have some aspect of common behavior which is defined by the inclusion of operations and properties from some common component portType.

646

647

648

Using the behaviors defined by the ScheduledResourceTermination portType as an example, the ServiceGroup may include the following MembershipContentRule:

649

650

651

```
<wsrf-sg:MembershipContentRule  
ContentElements="wsrf-rl:TerminationTime" />
```

652

653

654

This constrains the content of every ServiceGroupEntry to require a TerminationTime element. The service built on ServiceGroup should then require the following (which are not normative requirements of the ServiceGroup specification):

655

656

657

1. The corresponding member service also has the TerminationTime property in its resource properties document schema

658

659

2. The SetTerminationTime operation exists in the portType of the member service.

660

661

662

3. The WSDL of the member service includes the wsdl:Action attribute in the input element of this operation. The attribute value is <http://docs.oasis-open.org/wsrf/rlw-2/ScheduledResourceTermination/SetTerminationTimeRequest>.

663

These three requirements relate the content of the ServiceGroupEntry to behavior defined by the ScheduledResourceTermination portType.

664

---

## 665 **7 AppNotes Relating to Resource Metadata**

### 666 **7.1 Derived Resource Metadata**

667 Resource Metadata [**WS-ResourceMetadataDescriptor**] specifies a format for additional  
668 metadata which describes the resource properties of a WS-Resource. As with derived  
669 portTypes using “cut-and-paste” a similar process should be followed for the derived  
670 MetadataDescriptor document.

#### 671 **7.1.1 Best Practices and Examples**

672 A MetadataDescriptor component can optionally specialise one or more other  
673 MetadataDescriptor components. However, as with WSDL 1.1, there is no formal way to  
674 extend an existing MetadataDescriptor without copying its contents into the new  
675 MetadataDescriptor. In such cases the MetadataDescriptor component should contain the  
676 properties of the MetadataDescriptor component(s) it specializes in addition to the properties  
677 it already defines.

678 When performing this aggregation, it is important to remember that no two Property elements  
679 can share a {path} value. If a new MetadataDescriptor extends two other MetadataDescriptors  
680 that happen to share a Property {path}, the new descriptor must consolidate these duplicate  
681 Property definitions into one. The author of the new descriptor is free to decide which of the  
682 Property attributes and child elements will be included in the new definition, to prevent the  
683 scenario described above (because we are using cut-and-paste rather than formal  
684 inheritance, the author is not obliged to honour the restrictions of previous definitions).

685

686

687

688

## 8 Interoperability Considerations

689

### 8.1 Interoperability

690

In order to ensure interoperability of WSRF implementations, WSRF applications should use a document-literal binding to serialize the messages defined by the WSRF specifications.

691

692

### 8.2 Intermediary Changes of Namespace Prefixes

693

SOAP allows intermediaries to modify the XML namespace prefixes of messages passing through them. This may result in interoperability problems if attributes or text nodes in the message contain QName, as the QName prefixes in attributes and text nodes will not be recognized as such by the intermediary. For example, the following query uses the prefix 'pr' to refer to a component of the Printer properties in order to discover whether the printer supports the 'text/plain' MIME type:

694

695

696

697

698

699

```
<s11:Body>
  <wsrf-rp:QueryResourceProperties>
    <wsrf-rp:QueryExpression
      xmlns:pr=http://docs.oasis-open.org/wsrf/Printer.xsd
      Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
      contains( /* */ pr:mimeType, "text/plain" )
    </wsrf-rp:QueryExpression>
  </wsrf-rp:QueryResourceProperties>
</s11:Body>
```

700

701

702

703

704

705

706

707

708

#### 8.2.1 Best Practices and Examples

709

For cases where the prefix is in an XPath, the XPath expressions can be rewritten in such a way that they are protected against such intermediary behavior. For example, the following query uses the full namespace of the Printer property:

710

711

712

```
<s11:Body>
  <wsrf-rp:QueryResourceProperties>
    <wsrf-rp:QueryExpression
      Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
      contains( /* */ *[namespace-uri() =
      'http://docs.oasis-open.org/wsrf/Printer.xsd'
      and local-name() = 'mimeType'], "text/plain" )
    </wsrf-rp:QueryExpression>
  </wsrf-rp:QueryResourceProperties>
</s11:Body>
```

713

714

715

716

717

718

719

720

721

722

723

Alternatively, an application deployer might choose to use a different vendor's intermediary that does not alter the message prefixes.

724

---

725

## References

726

### 8.3 Normative

727

[WS-Addressing]

728

<http://www.w3.org/TR/ws-addr-core>

729

[WS-BaseFaults]

730

[http://docs.oasis-open.org/wsrf/wsrf-ws\\_base\\_faults-1.2-spec-os.pdf](http://docs.oasis-open.org/wsrf/wsrf-ws_base_faults-1.2-spec-os.pdf)

731

[WS-BaseNotification]

732

[http://docs.oasis-open.org/wsn/wsn-ws\\_base\\_notification-1.3-spec-pr-02.pdf](http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-pr-02.pdf)

733

734

[WS-Resource]

735

[http://docs.oasis-open.org/wsrf/wsrf-ws\\_resource-1.2-spec-os.pdf](http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf)

736

[WS-ResourceLifetime]

737

[http://docs.oasis-open.org/wsrf/wsrf-ws\\_resource\\_lifetime-1.2-spec-os.pdf](http://docs.oasis-open.org/wsrf/wsrf-ws_resource_lifetime-1.2-spec-os.pdf)

738

739

[WS-ResourceMetadataDescriptor]

740

[http://docs.oasis-open.org/wsrf/wsrf-ws\\_resource\\_metadata\\_descriptor-1.0-spec-cd-01.pdf](http://docs.oasis-open.org/wsrf/wsrf-ws_resource_metadata_descriptor-1.0-spec-cd-01.pdf)

741

742

[WS-ResourceProperties]

743

[http://docs.oasis-open.org/wsrf/wsrf-ws\\_resource\\_properties-1.2-spec-os.pdf](http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-os.pdf)

744

745

[WS-ServiceGroup]

746

[http://docs.oasis-open.org/wsrf/wsrf-ws\\_service\\_group-1.2-spec-os.pdf](http://docs.oasis-open.org/wsrf/wsrf-ws_service_group-1.2-spec-os.pdf)

747

748

[WS-Topics]

749

[http://docs.oasis-open.org/wsn/wsn-ws\\_topics-1.3-spec-pr-01.pdf](http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-pr-01.pdf)

750

### 8.4 Non-Normative

751

[WSRFPrimer]

752

<http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf>

753

[WSDM] Web Services Distributed Management

754

[http://www.oasis-](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm)

755

[open.org/committees/tc\\_home.php?wg\\_abbrev=wsdm](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm)

756

757

---

## Appendix A. Acknowledgments

758 The following individuals were members of the committee during the development of this  
759 document:

760

761 Mario Antonioletti (EPCC, The University of Edinburgh), Akhil Arora (Sun Microsystems), Tim  
762 Banks (IBM), Jeff Bohren (OpenNetwork), Fred Carter (AmberPoint), Martin Chapman  
763 (Oracle), Glen Daniels (Sonic Software), David De Roure (University of Southampton),  
764 Thomas Freund (IBM), John Fuller (Individual), Stephen Graham (IBM), Anish Karmarkar  
765 (Oracle), Hideharu Kato (Hitachi), David Levine (IBM), Paul Lipton (Computer Associates),  
766 Mark Little (Arjuna Technologies Limited), Lily Liu (WebMethods, Inc.), Tom Maguire (IBM),  
767 Susan Malaika (IBM), Mark Mc Keown (University of Manchester), David Martin (IBM),  
768 Samuel Meder (Argonne National Laboratory), Jeff Mischkin (Oracle), Roger Menday  
769 (Forschungszentrum Juelich GmbH), Bryan Murray (Hewlett-Packard), Mark Peel (Novell),  
770 Alain Regnier (Ricoh Company, Ltd.), Ian Robinson (IBM), Tom Rutt (Fujitsu), Mitsunori  
771 Satomi (Hitachi), Igor Sedukhin (Computer Associates), Hitoshi Sekine (Ricoh Company,  
772 Ltd.), Frank Siebenlist (Argonne National Laboratory), Alex Sim (Lawrence Berkeley National  
773 Laboratory), David Snelling (Fujitsu), Latha Srinivasan (Hewlett-Packard), Rich Thompson  
774 (IBM), Jem Treadwell (Hewlett-Packard), Steve Tuecke (Argonne National Laboratory),  
775 William Vambenepe (Hewlett-Packard), Katy Warr (IBM), Alan Weissberger (NEC  
776 Corporation), Pete Wenzel (SeeBeyond Technology Corporation), Kirk Wilson (Computer  
777 Associates) and Umit Yalcinalp (SAP).

778

## Appendix B. Revision History

779

Rev	Date (MM/DD/YYYY)	By Whom	What
0.1	11/07/2005	Katy Warr	Initial Creation based on AppNotes outline from Alan
0.2	16/01/2005	Katy Warr	Begin to bring in line with Primer and other docs for consistency and to remove duplication.
0.3	02/01/2005	Katy Warr	Updated base fault section
0.4	21/02/2005	Katy Warr	Minor changes for f2f
0.5	03/09/2005	Katy Warr	Some of the issues raised at the feb f2f: <ul style="list-style-type: none"> <li>- Add updates to do section so discussions aren't forgotten.</li> <li>- moved non-normative txt from WSRP section 4 to appnotes</li> <li>- corrections and minor amendments</li> <li>- improve RAP embodiments section</li> <li>- Remove QOS section</li> </ul>
0.6	05/16/2005	Katy Warr	Draft resolutions to the following issues: 22, 63, 52, 89, 95
wd-03	05/20/2005	Ian Robinson	Editorial consistency
pr-01	06/13/2005	Ian Robinson	Change status to PR draft
wd-04	10/18/2005	Roger Menday	Resolution to 103. Misc. edits. Replaced DiskDrive RP example with Printer. Integrated feedback.
wd-05	11/28/2005	Roger Menday	Integrated comments from Ian Banks, Jem Treadwell and Daniel Jemiolo.
wd-06	1/18/2006	Roger Menday	New sections on ServiceGroup and Resource Metadata.
wd-07	2/18/2006	Roger Menday	Integrated comments from Ian Robinson and Jem Treadwell.
cd-02	3/25/2005	Roger Menday	Inserted some post-vote edits from Kirk Wilson.

780

---

## 781 Appendix C. Notices

782 OASIS takes no position regarding the validity or scope of any intellectual property or other  
783 rights that might be claimed to pertain to the implementation or use of the technology  
784 described in this document or the extent to which any license under such rights might or might  
785 not be available; neither does it represent that it has made any effort to identify any such  
786 rights. Information on OASIS's procedures with respect to rights in OASIS specifications can  
787 be found at the OASIS website. Copies of claims of rights made available for publication and  
788 any assurances of licenses to be made available, or the result of an attempt made to obtain a  
789 general license or permission for the use of such proprietary rights by implementors or users  
790 of this specification, can be obtained from the OASIS Executive Director.

791

792 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
793 applications, or other proprietary rights which may cover technology that may be required to  
794 implement this specification. Please address the information to the OASIS Executive Director.

795

796 Copyright (C) OASIS Open (2006). All Rights Reserved.

797

798 This document and translations of it may be copied and furnished to others, and derivative  
799 works that comment on or otherwise explain it or assist in its implementation may be  
800 prepared, copied, published and distributed, in whole or in part, without restriction of any kind,  
801 provided that the above copyright notice and this paragraph are included on all such copies  
802 and derivative works. However, this document itself may not be modified in any way, such as  
803 by removing the copyright notice or references to OASIS, except as needed for the purpose  
804 of developing OASIS specifications, in which case the procedures for copyrights defined in  
805 the OASIS Intellectual Property Rights document must be followed, or as required to translate  
806 it into languages other than English.

807

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

810

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