



29 **Status:**

30 This document is published by this TC as a "public review draft". It is possible that it may  
31 change significantly during this process, but should nonetheless provide a stable  
32 reference for discussion and early adopters' implementations.

33  
34 Committee members should send comments on this specification to the [wsrf@lists.oasis-](mailto:wsrflists@lists.oasis-open.org)  
35 [open.org](mailto:wsrflists@lists.oasis-open.org) list. Others should subscribe to and send comments to the [wsrf-](mailto:wsrflists@lists.oasis-open.org)  
36 [comment@lists.oasis-open.org](mailto:wsrflists@lists.oasis-open.org) list. To subscribe, send an email message to [wsrf-](mailto:wsrflists@lists.oasis-open.org)  
37 [comment-request@lists.oasis-open.org](mailto:wsrflists@lists.oasis-open.org) with the word "subscribe" as the body of the  
38 message.

39  
40 For information on whether any patents have been disclosed that may be essential to  
41 implementing this specification, and any offers of patent licensing terms, please refer to  
42 the Intellectual Property Rights section of the WSRF TC web page ([http://www.oasis-](http://www.oasis-open.org/committees/wsrflists/)  
43 [open.org/committees/wsrflists/](http://www.oasis-open.org/committees/wsrflists/)).

## Table of Contents

45	<b>1 INTRODUCTION</b> .....	<b>5</b>
46	1.1 GOALS AND REQUIREMENTS .....	6
47	1.1.1 Requirements .....	6
48	1.1.2 Non-Goals.....	6
49	1.2 TERMINOLOGY .....	7
50	1.3 NAMESPACES .....	8
51	<b>2 TERMINOLOGY AND CONCEPTS</b> .....	<b>9</b>
52	<b>3 EXAMPLE</b> .....	<b>10</b>
53	3.1 THE OPERATINGSYSTEM PORTTYPE.....	10
54	3.2 OPERATING SYSTEM PROPERTIES.....	11
55	3.2.1 Operating System Resource Property definitions .....	11
56	3.2.2 Identification Property definitions.....	11
57	3.2.3 MetadataDescriptor for Identification portType .....	12
58	3.2.4 MetadataDescriptor for OperatingSystem portType .....	13
59	<b>4 LOGICAL MODEL FOR METADATA</b> .....	<b>16</b>
60	<b>5 INFORMATION MODEL FOR WS-RESOURCE METADATA</b> .....	<b>17</b>
61	<b>6 DEFINITIONS COMPONENT</b> .....	<b>18</b>
62	6.1 METADATADESCRIPTOR COMPONENTS WITHIN A DEFINITIONS COMPONENT .....	19
63	<b>7 METADATADESCRIPTOR COMPONENT</b> .....	<b>20</b>
64	7.1 PROPERTIES COMPONENT OF A METADATADESCRIPTOR .....	21
65	<b>8 PROPERTY COMPONENT</b> .....	<b>22</b>
66	8.1 XML SCHEMA VALUE SPACE AND {VALIDVALUES} .....	24
67	8.2 VALIDVALUES .....	25
68	8.3 VALIDVALUERANGE.....	26
69	8.4 STATICVALUES .....	28
70	8.5 INITIALVALUES .....	29
71	<b>9 DOCUMENTATION COMPONENT</b> .....	<b>31</b>
72	<b>10 OBTAINING A METADATADESCRIPTOR DOCUMENT</b> .....	<b>31</b>
73	10.1 EXTENDING WSDL 1.1 PORTTYPE.....	31
74	10.2 USING RESOURE PROPERTY ELEMENTS TO EXPOSE METADATADESCRIPTORS.....	32
75	<b>11 REFERENCES</b> .....	<b>34</b>

76	11.1	NORMATIVE .....	34
77	11.2	NON-NORMATIVE.....	34
78	<b>APPENDIX A. ACKNOWLEDGMENTS .....</b>		<b>35</b>
79	<b>APPENDIX B. XML SCHEMA FOR WS-RESOURCEMETADATADESCRIPTOR.....</b>		<b>36</b>
80	<b>APPENDIX C. REVISION HISTORY.....</b>		<b>45</b>
81	<b>APPENDIX D. NOTICES.....</b>		<b>46</b>
82			

## 83 1 Introduction

84 In the WS-Resource Framework [WSRF], elements of a WS-Resource's state are exposed to  
85 third party requestors through an XML document. The XML document associated with a WS-  
86 Resource is called a *resource properties document*. The resource properties document is a  
87 projection of the WS-Resource's state (not all of the elements of a WS-Resource's state are  
88 exposed through the resource properties document). An individual element of state contained in a  
89 resource properties document is called a *resource property*. Access to the resource properties  
90 document is governed by Web services operations defined in the WS-Resource Properties [WS-  
91 Resource Properties] specification. These operations generically allow for get, set and query of  
92 resource properties.

93

94 In many cases, some of the resource properties exposed through the resource properties  
95 document are not accessible through every operation defined in WS-ResourceProperties. The  
96 most common case of this is a resource property that is "read-only" implying that a requestor may  
97 not use Web services message exchanges (such as the WS-ResourceProperties  
98 SetResourceProperties operation) to change the value of the resource property. Clearly, an  
99 implementation of a WS-Resource is likely to return a fault message if a requestor attempts to  
100 change the value of a "read-only" resource property. However, in the absence of additional  
101 metadata, there is no standard means by which the requestor can determine a priori that the  
102 resource property was not modifiable.

103

104 We refer to the concept of a WS-Resource Metadata Descriptor to describe a unit of metadata  
105 information associated with the interface components of a WS-Resource. We describe an  
106 information model that outlines the components of metadata and their relationships to interface  
107 description artifacts such as WSDL 1.1 portTypes and resource properties document schema  
108 definitions.

109

110 A WS-Resource Metadata Descriptor serves multiple purposes. The first is to provide additional  
111 information about the resource properties of a WS-Resource. For instance, indicating whether a  
112 resource property is changeable using Web services message exchanges such as the  
113 SetResourceProperties operation described in the WS-ResourceProperties specification [WS-  
114 ResourceProperties]. This aspect of the MetadataDescriptor is associated with the interface of  
115 the WS-Resource, and would not vary between different implementations of the interface.  
116 Information in the MetadataDescriptor provides clients of a WS-Resource the potential for greater  
117 understanding of the behavior of that WS-Resource.

118

119 The second purpose is to provide information about the value restrictions of the resource  
120 properties in the resource properties document for the WS-Resource. This additional information  
121 may be associated with implementations of the interface as well as with the WSDL interface  
122 definition.

123 The single portType that describes the manageability interface for a manageable resource type is  
124 derived from various other manageability portTypes. With WSDL 1.1, physically including, using  
125 copy and paste, the operations from each of these portTypes into the definition of the most  
126 derived portType achieves this inheritance. Each of the portTypes from which a manageable  
127 resource's portType is derived may also have a MetadataDescriptor to augment its description.  
128 Each of the portTypes may have an optional attribute information item that references a  
129 MetadataDescriptor component by its namespace qualified name (QName).

130

131 This document standardizes the form of the WS-Resource MetadataDescriptor that contains  
132 metadata information about a WS-Resource's interface so that clients of that interface may  
133 reason about implementations of the interface at both design time and run time. The syntax of a  
134 preferred XML serialization of the information model is also described.

135

136 A portion of the Global Grid Forum's "Open Grid Services Infrastructure (OGSI) Version 1.0"  
137 specification [OGSI] inspired many of the concepts expressed in this document.

138

## 139 **1.1 Goals and Requirements**

140 The goal of this document is to define the terminology, concepts, information model and XML  
141 definitions needed to express the metadata requirements of WS-Resources, as defined by the  
142 [WS-Resource] specification.

143

### 144 **1.1.1 Requirements**

145 In meeting this goal, the specification MUST address the following specific requirements:

- 146 • Define an information model representing metadata about resource properties associated  
147 with a WS-Resource interface.
- 148 • Define a standard annotation for associating metadata descriptions with other description  
149 artifacts of the WS-Resource, particularly its WSDL 1.1 portType and its resource  
150 properties document definition.
- 151 • Define the standard schema for representing the aspects of the information model.

152

### 153 **1.1.2 Non-Goals**

154 The following topics are outside the scope of this specification:

- 155 • It is not an objective of this specification to define new message exchanges required to  
156 access the metadata from a WS-Resource.
- 157 • It is not an objective of this specification to describe the means required to store the  
158 metadata for a WS-Resource.

## 159 1.2 Terminology

160 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",  
161 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be  
162 interpreted as described in [RFC 2119].

163

164 When describing abstract data models, this specification uses the notational convention used by  
165 the [XML Infoset]. Specifically, abstract property names always appear in square brackets (e.g.,  
166 [some property]).

167

168 This specification uses a notational convention, referred to as "Pseudo-schemas" in a fashion  
169 similar to the WSDL 2.0 Part 1 specification [WSDL 2.0]. A Pseudo-schema uses a BNF-style  
170 convention to describe attributes and elements:

171 `?' denotes optionality (i.e. zero or one occurrences),

172 `\*' denotes zero or more occurrences,

173 `+' one or more occurrences,

174 `[ ]' and `{ }' are used to form groups,

175 `/' represents choice.

176

177 Attributes are conventionally assigned a value which corresponds to their type, as defined in the  
178 normative schema.

179

```
180 <!-- sample pseudo-schema -->  
181 <element  
182     required_attribute_of_type_QName="xs:QName"  
183     optional_attribute_of_type_string="xs:string"? >  
184     <required_element />  
185     <optional_element />?  
186     <one_or_more_of_these_elements />+  
187     [ <choice_1 /> | <choice_2 /> ]*  
188 </element>
```

189

190

191

192

193

194

195

## 1.3 Namespaces

196

The following namespaces are used in this document:

Prefix	Namespace
xs	<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>
wsdl	<a href="http://schemas.xmlsoap.org/wsdl">http://schemas.xmlsoap.org/wsdl</a>
wsrf-rp	<a href="http://docs.oasis-open.org/wsrp/rp-2">http://docs.oasis-open.org/wsrp/rp-2</a>
wsrmd	<a href="http://docs.oasis-open.org/wsrp/rmd-1">http://docs.oasis-open.org/wsrp/rmd-1</a>

197

## 198 **2 Terminology and Concepts**

199 The following definitions outline the terminology and usage in this specification. This section gives  
200 only brief description of these terms.

201

202 Metadata:

- 203 • Data about data. In practice, metadata comprises a structured set of descriptive elements  
204 to describe an information resource. Currently, only a WS-Resource's resource  
205 properties have metadata definitions.  
206

207 MetadataDescriptor:

- 208 • A unit of containment for resource property metadata for a WS-Resource's interface;  
209 property metadata is defined by zero or more Property elements.  
210

211 MetadataDescriptor Document:

- 212 • An XML instance document whose root is a Definitions element from the wsrmd  
213 namespace. This document contains definitions for zero or more MetadataDescriptor  
214 components.

## 215 **3 Example**

216 In the following example there are “Operating System” WS-Resources whose values are  
217 projected from the implementation of the *OperatingSystem* portType.

### 218 **3.1 The OperatingSystem portType**

219 The *OperatingSystem* portType defines operations and a resource properties document that  
220 describes the Web services interface to operating system resource instances. As well as  
221 providing a mechanism for interacting with the operating system itself, this portType also  
222 describes properties, which represent the hardware of the underlying machine on which the  
223 operating system is running.

224

225 The *OperatingSystem* portType is derived from various other manageability portTypes – these  
226 illustrations use some of the function from the *Identification* manageability portType. As is  
227 required with WSDL 1.1, this derivation is achieved by physically including the definitions from  
228 each of these portTypes in the definition of the *OperatingSystem* portType. This “cut-and-paste”  
229 system of derivation is discussed further in [AppNotes].

230

231 The *OperatingSystem* portType is sketched as follows:

232

```
233 (01) ... xmlns:os="http://example.com/ns/OperatingSystem"  
234 (02) ... xmlns:id="http://example.com/ns/Identification"  
235 (03) ...  
236 (04) <portType name="OperatingSystem"  
237 (05)     wsrp-rp:ResourceProperties="os:OSResourceProperties"  
238 (06)     ..wsrmd:Descriptor="os:OperatingSystemMetadataDescriptor"  
239 (07)     ..wsrmd:DescriptorLocation="http://example.com/OperatingSystem.wsrmd" >  
240 (08) ...  
241 (09) </portType>
```

242

243 Line (04) contains a portType declaration for a portType named OperatingSystem in the  
244 namespace corresponding to the os: namespace prefix declaration.

245

246 Line (05) indicates the global XML element declaration of the root element of the resource  
247 properties document associated with any WS-Resource whose Web service implements the  
248 os:OperatingSystem portType.

249

250 Line (06) identifies that a MetadataDescriptor has been defined for this interface identified by the  
251 QName os:OperatingSystemMetadataDescriptor.

252

253 Line (07) - (07) indicate that information about MetadataDescriptors in the namespace  
254 corresponding to the os: namespace prefix declaration can be found by dereferencing  
255 <http://example.com/OperatingSystem.wsrmd>.

## 256 3.2 Operating System Properties

257 In the following example we define a subset of the resource properties of the operating system.  
258 Following that are the MetadataDescriptors for the Identification and Operating System  
259 portTypes.

### 260 3.2.1 Operating System Resource Property definitions

```
261 (10) ...  
262 (11)   xmlns:os="...  
263 (12)  
264 (13) <element name="OSResourceProperties">  
265 (14)   <complexType>  
266 (15)     <sequence>  
267 (16)       <element ref="id:ResourceType" minOccurs="0" maxOccurs="1"/>  
268 (17)       <element ref="id:ResourceID" minOccurs="0" maxOccurs="1"/>  
269 (18)       <element ref="os:numberOfProcesses" minOccurs="0" maxOccurs="1"/>  
270 (19)       <element ref="os:totalSwapSpaceSize" minOccurs="0" maxOccurs="1"/>  
271 (20)       <element ref="os:processor" minOccurs="1" maxOccurs="unbounded"/>  
272 (21)     </sequence>  
273 (22)   </complexType>  
274 (23) </element>  
275 (24)  
276 (25) <element name="numberOfProcesses" type="xsd:int" />  
277 (26) <element name="totalSwapSpaceSize" type="xsd:unsignedLong" />  
278 (27) <element name="processor" type="xsd:string" />  
279 (28)
```

### 280 3.2.2 Identification Property definitions

281

```
282 (29) <element name="IdentificationResourceProperties">  
283 (30)   <complexType>  
284 (31)     <sequence>  
285 (32)       <element ref="ResourceType" minOccurs="0" maxOccurs="1"/>  
286 (33)       <element ref="ResourceID" minOccurs="0" maxOccurs="1"/>  
287 (34)     </sequence>  
288 (35)   </complexType>  
289 (36) </element>
```

290  
291  
292

```
(37)  
(38) <element name="ResourceType" type="xs:string" />  
(39) <element name="ResourceID" type="xs:string" />
```

### 293 3.2.3 MetadataDescriptor for Identification portType

294 An example MetadataDescriptor document for the Identification portType is included below. For  
295 the purposes of this example, this MetadataDescriptor document will be located at the URL  
296 <http://example.com/metadataDescriptors/Identification.wsrmtd>.

297

298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315

```
(40)  
(41) <Definitions  
(42)     xmlns="http://docs.oasis-open.org/wsr/1.0/"  
(43)     xmlns:id="http://example.com/ns/Identification"  
(44)     targetNamespace="http://example.com/ns/Identification">  
(45)   <MetadataDescriptor  
(46)     name="IdentificationMetadataDescriptor"  
(47)     interface="id:Identification"  
(48)     wsdlLocation="http://example.com/ns/Identification  
(49)       http://example.com/wsd/Identification.wsdl" >  
(50)   <Property name="id:ResourceID"  
(51)     mutability="constant"  
(52)     modifiability="read-only" />  
(53)   <Property name="id:ResourceType"  
(54)     mutability="constant"  
(55)     modifiability="read-only" />  
(56)   </MetadataDescriptor>  
(57) </Definitions>
```

316

317 Line (41) contains a Definitions element defining MetadataDescriptor elements for the target  
318 namespace identified in line (44).

319

320 There is one MetadataDescriptor element child of this Definitions element (lines (45)–(56)).

321 The name of the MetadataDescriptor is contained in line (46). This together with the namespace  
322 prefix declaration in line (43) corresponding to the targetNamespace of the Definitions element  
323 means the QName of the MetadataDescriptor is id:IdentificationMetadataDescriptor.

324

325 Line (47) identifies the QName of the portType (interface) with which this MetadataDescriptor is  
326 associated. The location of WSDL for the Identification portType is expressed in the wsdlLocation  
327 attribute in lines (48)–(49). This follows the pattern of the wsdl:wsdlLocation attribute defined in  
328 the WSDL 2.0 specification [WSDL 2.0].

329

330 Lines (50)-(55) show two Property elements containing metadata information about resource  
331 properties defined in the resource properties document for the Identification portType. Lines (50)-  
332 (52) contain the first Property element that references the QName of the id:ResourceID resource  
333 property. Line (51) indicates that the id:ResourceID resource property element will always have a  
334 constant value. Line (52) states that the id:ResourceID resource property is read-only, meaning  
335 that it cannot be changed by a requestor using Web services message exchanges such as the  
336 SetResourceProperties operation as defined in WS-ResourceProperties.

337

338 The second Property element, in lines (53)-(55), references the QName of the id:ResourceType  
339 resource property in line (53). This resource property element has the same metadata attributes  
340 as id:ResourceIdentifier.

### 341 **3.2.4 MetadataDescriptor for OperatingSystem portType**

342 For the purposes of this example the MetadataDescriptor document for the OperatingSystem  
343 portType is found at <http://example.com/metadataDescriptors/OperatingSystem.wsrm>.  
344

345 The contents of the MetadataDescriptor for the OperatingSystem portType appear as follows:

346

```
347 (58)      <Definitions
348 (59)          xmlns="http://docs.oasis-open.org/wsrf/rmd-1"
349 (60)          xmlns:id="http://example.com/ns/Identification"
350 (61)          xmlns:os="http://example.com/ns/OperatingSystem"
351 (62)          targetNamespace="http://example.com/ns/OperatingSystem">
352 (63)      <MetadataDescriptor
353 (64)          name="OperatingSystemMetadataDescriptor"
354 (65)          interface="os:OperatingSystem"
355 (66)          wsdlLocation="http://example.com/ns/OperatingSystem
356 (67)              http://example.com/wsdl/OperatingSystem.wsdl">
357 (68)      <Property name="id:ResourceID"
358 (69)          mutability="constant"
359 (70)          modifiability="read-only" />
360 (71)      <Property name="id:ResourceType
361 (72)          mutability="constant"
362 (73)          modifiability="read-only">
363 (74)          <ValidValues>
364 (75)              <id:ResourceType>SuSELinux</id:ResourceType>
365 (76)              <id:ResourceType>IBMzOS</id:ResourceType>
366 (77)              <id:ResourceType>MicrosoftWindows_XP</id:ResourceType>
367 (78)          </ValidValues>
368 (79)      </Property>
```

```

369 (80)         <Property name="os:numberOfProcesses"
370 (81)             mutability="mutable"
371 (82)             modifiability="read-only" />
372 (83)         <Property name="os:processor"
373 (84)             mutability="constant"
374 (85)             modifiability="read-only">
375 (86)             <ValidValues>
376 (87)                 <os:processor>Pentium Family</os:processor>
377 (88)                 <os:processor>Power PC 750</os:processor>
378 (89)                 <os:processor>68xxx Family</os:processor>
379 (90)                 <os:processor>PA-RISC Family</os:processor>
380 (91)                 <os:processor>Alpha Family<os:processor>
381 (92)                 <os:processor>IBM390 Family</os:processor>"
382 (93)                 <os:processor>G5</os:processor>
383 (94)                 <os:processor>AMD<os:processor>
384 (95)             </ValidValues>
385 (96)         </Property>
386 (97)     </MetadataDescriptor>
387 (98) </Definitions>

```

388 Lines (58) to (98) contain a Definitions element for the <http://example.com/ns/OperatingSystem>  
389 namespace.

390

391 Lines (63) to (98) contain the definition of the MetadataDescriptor with QName  
392 `os:OperatingSystem MetadataDescriptor`.

393

394 Line (65) indicates that this MetadataDescriptor corresponds to the OperatingSystem portType.

395 Lines (66)-(67) gives the location of the WSDL document that defines elements associated with  
396 the namespace URI associated with the `os:` prefix (i.e. the WSDL definitions element that defines  
397 the OperatingSystem portType).

398

399 Lines (68)-(96) contains the four properties described in this MetadataDescriptor example.

400 Lines (68) –(70) contain the ResourceID Property element copied from the  
401 `id:IdentificationMetadataDescriptor` describing the Identification portType.

402 Lines (71)-(79) contain the Property element describing the `id:ResourceType` resource property  
403 from the Identification portType. Lines (74)-(78) contain the set of ValidValues that the  
404 `id:ResourceType` resource property may contain.

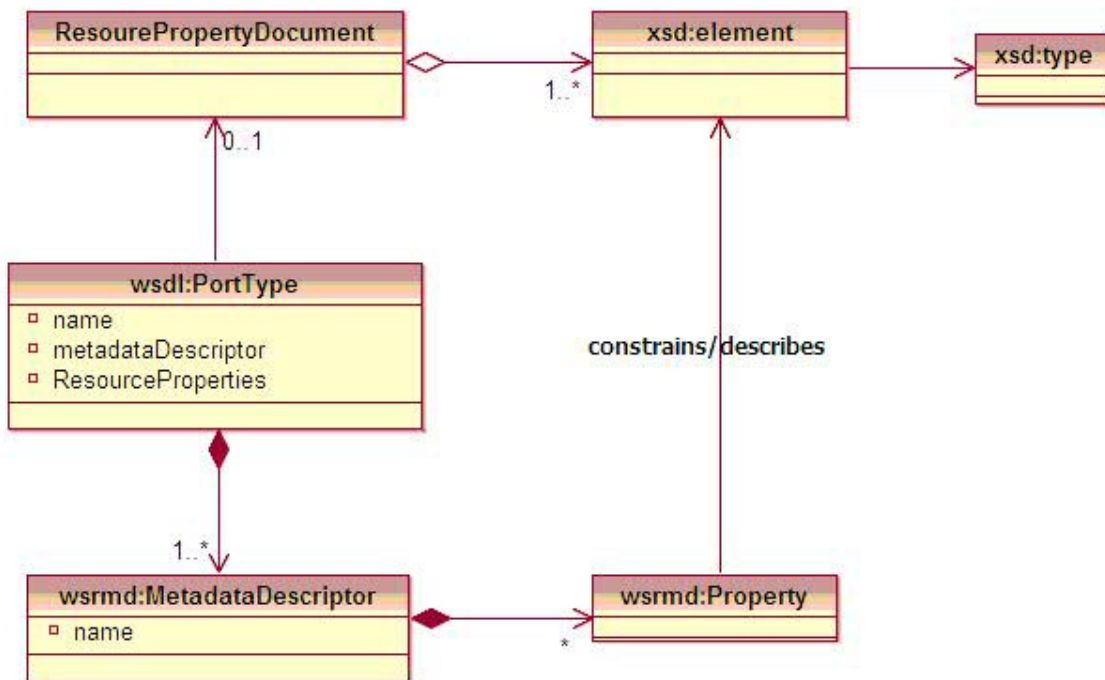
405

406 Lines (80)-(82) contain the `os:numberOfProcesses` Property element which references the  
407 QName of the `os:numberOfProcesses` resource property. Line (81) indicates that the value of the  
408 `os:numberOfProcesses` may change over time. Line (82) indicates that, the

409 os:numberOfProcesses can not be changed by a requestor using Web services message  
410 exchanges such as the SetResourceProperties operation as defined in WS-ResourceProperties  
411 [WS-ResourceProperties].  
412 The next Property element references the os:processor. The modifiability and mutability values  
413 indicate that the property is static – it will not change during the resource’s lifetime. Lines (87)-  
414 (94) describe valid values for the os:processor.

415 **4 Logical Model for Metadata**

416 The following figure shows a logical model depicting the relationship between the various  
 417 elements of metadata description and those elements the metadata describes.



418

419 *Figure 1 Logical Model of WS-Resource MetadataDescriptor*

420

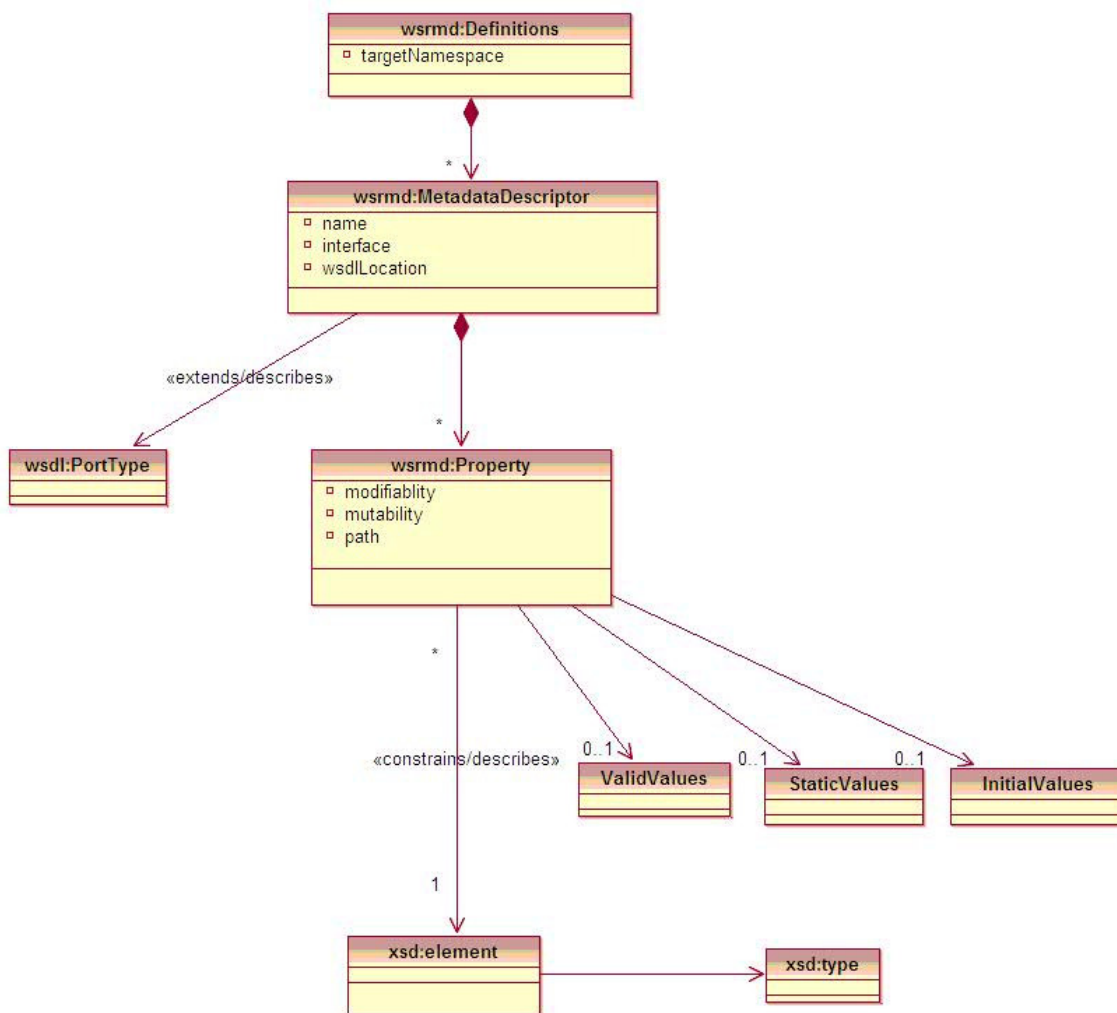
421 In our model, the unit of metadata containment is referred to as a *MetadataDescriptor*. A  
 422 *MetadataDescriptor* is used to describe aspects of a WS-Resource's interface, particularly those  
 423 elements associated with the WS-Resource's WSDL 1.1 portType.

424

425 A *MetadataDescriptor* contains metadata describing and/or constraining resource property  
 426 elements as contained within a WS-Resource's Resource Properties document type definition.  
 427 Each resource property element is defined as an XML Schema global element, in some  
 428 namespace.

429 **5 Information Model for WS-Resource Metadata**

430 This section describes the information model for metadata describing/constraining the resource  
 431 properties of WS-Resources. The model is a simple hierarchy – each WSDL portType *references*  
 432 a Resource Metadata Descriptor document, and that file *contains* a Definitions element, which  
 433 *contains* MetadataDescriptors, which *contain* Property elements. A UML diagram of this model is  
 434 shown in the following figure:



435  
 436  
 437

Figure 2 - Information Model for WS-Resource Metadata Descriptor

438 We describe the Definitions, MetadataDescriptor, and Property components in the following  
 439 sections.

## 6 Definitions Component

The Definitions component is a container for a set of MetadataDescriptor components (see section 7). The Definitions component defines a targetNamespace which forms the {namespace} property of all components it contains.

The properties of a Definitions component are as follows:

- {targetNamespace} a namespace URI that applies as the {namespace} property to all [child] components.
- {metadataDescriptors} a set of zero or more MetadataDescriptor components.

The following is an XML representation of the Definitions component:

```
<Definitions
  targetNamespace="xs:anyURI"
  {anyAttribute}* >

  <documentation />?
  <MetadataDescriptor /> *

</Definitions>
```

The Definitions *element information item* has the following Infoset properties:

- A [local name] of "Definitions".
- A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- one or more *attribute information items* amongst its [attributes] as follows:
  - A REQUIRED targetNamespace *attribute information item*
    - The value of this attribute information item contains a URI that defines the {namespace} property of all [child] components.
    - The type of the targetNamespace *attribute information item* is xs:anyURI.
  - Zero or more namespace qualified *attribute information items*. The [namespace name] of such *attribute information items* MUST NOT be "http://docs.oasis-open.org/wsrf/rmd-1".
- Zero or more *element information items* amongst its [children], in order as follows:
  - An OPTIONAL documentation *element information item* (See section 7).
  - Zero or more MetadataDescriptor *element information items* (See section 6.1).

477 **6.1 MetadataDescriptor components within a Definitions**  
478 **component**

479 All MetadataDescriptor components defined in a given namespace MUST appear as [children] of  
480 a Definitions component with {targetNamespace} value the same URI as that namespace. All  
481 MetadataDescriptor components MUST be uniquely named, implying that the {name} property of  
482 the MetadataDescriptor component MUST be unique amongst the {metadataDescriptors} of a  
483 Definitions component.

## 484 7 MetadataDescriptor Component

485 The MetadataDescriptor component is a container for a set of metadata descriptions and  
486 constraints on a WS-Resource. The MetadataDescriptor component contains additional  
487 information that describes or constrains various aspects of a WS-Resource. For example, it  
488 provides additional information about the interface of the WS-Resource relevant to the  
489 management of the resource. In particular, it allows tools and applications, such as management  
490 applications, to be able to reason in detail about the WS-Resource both at runtime and at  
491 development time when no instances of the WS-Resource are available.

492

493 The properties of a MetadataDescriptor component are as follows:

- 494 • {name} a name of a MetadataDescriptor component.
- 495 • {namespace} a namespace URI of the MetadataDescriptor component.
- 496 • {QName} a combination of the {name} and {namespace} of the MetadataDescriptor  
497 component.
- 498 • {interface} a QName identifying a Web services interface definition with which this  
499 MetadataDescriptor is associated.
- 500 • {wsdlLocation} a set of URI pairs, each pair associating a namespace URI with a URL of  
501 a document containing a WSDL definition of that namespace. This is a similar  
502 mechanism to that used in WSDL 2.0 [WSDL2.0].
- 503 • {properties} A set of zero or more Property components

504

505 The following is an XML representation of the MetadataDescriptor component:

506

```
507 <MetadataDescriptor  
508     name="xs:NCName"  
509     interface="xs:QName"  
510     wsdlLocation="list of xs:anyUri"?  
511     {anyAttribute}* >  
512   <documentation /> ?  
513   <Property /> *  
514   {any}*  
515 </MetadataDescriptor>
```

516

517

518 The MetadataDescriptor *element information item* has the following Infoset properties:

- 519 • A [local name] of "MetadataDescriptor" .
- 520 • A [namespace name] of "http://docs.oasis-open.org/wsr/rmd-1".

- 521
- two or more *attribute information items* amongst its [attributes] as follows:
    - 522 ○ A REQUIRED name attribute information item
      - 523 ▪ The value of this *attribute information item* contains the name of this
      - 524 MetadataDescriptor component.
      - 525 ▪ The type of the name *attribute information item* is xs:NCName.
    - 526 ○ A REQUIRED interface attribute information item
      - 527 ▪ The value of this *attribute information item* contains a QName of a WSDL
      - 528 1.1 portType element or WSDL 2.0 interface element associated with this
      - 529 MetadataDescriptor component.
      - 530 ▪ The type of the interface *attribute information item* is xs:QName.
    - 531 ○ An OPTIONAL wsdlLocation attribute information item
      - 532 ▪ The value of this *attribute information item* contains a list of pairs of
      - 533 URIs; where the first URI of the pair, which MUST be an absolute URI as
      - 534 defined in [URI], indicates a WSDL namespace name, and, the second a
      - 535 hint as to the location of a WSDL document defining WSDL components
      - 536 for that namespace name. The second URI of a pair MAY be absolute or
      - 537 relative.
      - 538 ▪ The type of the wsdlLocation *attribute information item* is list of
      - 539 xs:anyURI.
    - 540 ○ Zero or more namespace qualified *attribute information items*. The [namespace
    - 541 name] of such *attribute information items* MUST NOT be "http://docs.oasis-
    - 542 open.org/wsrf/rmd-1".
  - 543 • Zero or more *element information items* amongst its [children], in order as follows:
    - 544 ○ An OPTIONAL documentation *element information item* (See section 9).
    - 545 ○ Zero or more Property *element information items* (See section 8)
    - 546 ○ Zero or more namespace-qualified *element information items*. The [namespace
    - 547 name] of such *element information items* MUST NOT be "http://docs.oasis-
    - 548 open.org/wsrf/rmd-1".

## 549 7.1 Properties component of a MetadataDescriptor

550 The {properties} of a MetadataDescriptor contains a set of Property components, defining  
551 additional metadata and constraints on resource property elements (and attributes) associated  
552 with a MetadataDescriptor. The definition of a Property component's scope is contained in  
553 Section **Error! Reference source not found.**

## 8 Property Component

554

555 The Property component is a container for a set of metadata descriptions and constraints on a  
556 specific Resource Property element or attribute thereof. The properties of a Property component  
557 are as follows:

- 558 • {name} an identifier of the XML element to which the Property component applies. This is  
559 defined as a resource property's QName.
- 560 • {mutability} an xs:string enumeration of "constant", "appendable", or "mutable".
- 561 • {modifiability} an xs:string enumeration of "read-only" or "read-write".
- 562 • {subscribability} an xs:boolean indicating, if true, that the Resource Property element  
563 associated with the {name} can be the target of a subscription.
- 564 • {validValues} optional choice of one ValidValues component or one ValidValueRange  
565 component.
- 566 • {staticValues} optional choice of one StaticValues component.
- 567 • {initialValues} optional choice of one InitialValues component.
- 568 • {attributes} zero or more Attribute components.

569

570 A Property component MAY also contain additional "extension" components added using the  
571 extensibility mechanism defined by this specification. The following is an XML representation of  
572 the Property component:

573

574

```
<Property
  name="xs:QName"
  mutability="[constant|appendable|mutable]" ?
  modifiability="[read-only|read-write]" ?
  subscribability="xs:boolean" ?
  {anyAttribute}* >

  <documentation />?
  [ <ValidValues> {any}* </ValidValues> |
    <ValidValueRange
      lowerBound="xs:anySimpleType"? upperBound="xs:anySimpleType"?
    /> ] ?
  <StaticValues> {any}* </StaticValues> ?
  <InitialValues> {any}* </InitialValues> ?
  {any}*
</Property>
```

580

581

582

583

584

585

586

587

588

589

590

591 The Property *element information item* has the following Infoset properties:

- 592       • A [local name] of "Property" .
- 593       • A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- 594       • one or more *attribute information items* amongst its [attributes] as follows:
- 595           ○ A REQUIRED name attribute information item
- 596               ▪ The value of this *attribute information item* MUST contain a QName of
- 597               the Resource Property (an XML Schema global element definition)
- 598               contained within the Resource Properties document associated with the
- 599               portType or interface identified by {interface}. The Resource Property
- 600               element MUST conform to the requirements specified for Resource
- 601               Property declarations in WS-ResourceProperties.
- 602           ○ An OPTIONAL mutability attribute information item
- 603               ▪ The value of this *attribute information item* expresses how the value of
- 604               the {name} can change over time.
- 605               ▪ The type of the mutability *attribute information item* is an xs:string
- 606               restricted to the following enumeration:
- 607                   • "constant"
- 608                               The values of the {name} MUST NOT change after WS-
- 609                               Resource creation.
- 610                   • "mutable"
- 611                               The values of the {name} MAY change at any time during
- 612                               the lifetime of the WS-Resource. Existing values MAY be
- 613                               removed and new values MAY be added.
- 614                   • "appendable"
- 615                               The values of the {name} MAY have new values added
- 616                               during the lifetime of the WS-Resource. Once added those
- 617                               values MUST NOT be removed.
- 618               ▪ If the mutability *attribute information item* is not defined, the value of the
- 619               mutability property is "unknown".
- 620           ○ An OPTIONAL modifiability *attribute information item*
- 621               ▪ The value of this *attribute information item* indicates whether a requestor
- 622               can modify the value of the {name}.
- 623               ▪ The type of the modifiability *attribute information item* is an xs:string
- 624               restricted to the following enumeration:
- 625                   • "read-only" – The value of the {name} can not be changed by
- 626                               Web services message exchanges such as the
- 627                               SetResourceProperty message as defined in WS-
- 628                               ResourceProperties.
- 629                   • "read-write" – The value of the {name} MAY be changed by Web
- 630                               services message exchanges such as the SetResourceProperty

- 631 message as defined in WS-ResourceProperties. Note - If the  
632 value of the modifiability *attribute information item* is “read-write”  
633 then the value of the mutability *attribute information item* MUST  
634 NOT be “constant”.
- 635     ▪ If the modifiability attribute information item is not defined, the value of  
636     the modifiability property is “unknown”.
  - 637     ○ An OPTIONAL subscribability *attribute information item*
    - 638         ▪ The value of this *attribute information item* expresses whether the  
639         Resource Property element associated with the {name} can be the target  
640         of a subscription. The default value is “false”. Note: The actual  
641         subscription semantics are dependent on whatever notification  
642         mechanism, if any, (such as WS-BaseNotification [WS-BaseNotification])  
643         is supported.
  - 644     ○ Zero or more namespace qualified *attribute information items*. The [namespace  
645     name] of such *attribute information items* MUST NOT be "http://docs.oasis-  
646     open.org/wsrf/rmd-1".
  - 647     • Zero or more *element information items* amongst its [children], in order as follows:
    - 648         ○ An OPTIONAL documentation *element information item* (See section 9).
    - 649         ○ An OPTIONAL *element information item* from among the following:
      - 650             ○ A ValidValues element information item (See section 8.2)
      - 651             ○ A ValidValueRange element information item (See section 8.3)
      - 652             ○ An OPTIONAL StaticValues element information item (See section 8.4)
      - 653             ○ An OPTIONAL InitialValues element information item (See section 8.5)
    - 654         ○ Zero or more namespace-qualified *element information items*. The [namespace  
655         name] of such *element information items* MUST NOT be "http://docs.oasis-  
656         open.org/wsrf/rmd-1".

## 657 **8.1 XML Schema value space and {validValues}**

658 When creating a resource property (ie defining an XML Global Element), the XML Schema  
659 designer defines the semantic of the property and uses XML Schema to express the value space  
660 of the resource property (based on the semantics of the property) and all of its descendant  
661 *element information items* and *attribute information items*. This is a different concept from what is  
662 expressed by defining the {validValues}. When specifying {validValues} in a metadata description,  
663 one does not redefine the semantic of the {name} nor its value space. Specifying {validValues}  
664 expresses constraints on the value space that are appropriate for the specific use of the {name}.  
665 This distinction should guide designers in deciding whether to use XML Schema mechanisms or  
666 a metadata description to restrict value space of a {name}. The value space defined by  
667 {validValues} for a {name} MUST be contained within the XML Schema definition of the {name}.

## 668 8.2 ValidValues

669 The purpose of the ValidValues component is to restrict the set of valid values that a [parent]  
670 Property component's {name} may contain.

671

672 If the {validValues} of a Property component is not empty, and contains a ValidValues description,  
673 then any Web service that implements the portType or interface identified by {interface} MUST  
674 ensure that the value(s) of the {name} of the [parent] Property component MUST correspond to  
675 one of the values enumerated within the set of {validValues}.

676

677 Note: because the child *element information items* of a ValidValues *element information item* are  
678 XML fragments, it is not required that these fragments be validated (processContents is "skip"). .

679 The properties of a ValidValues component are as follows.

680

- 681 • {values} zero or more XML fragments that correspond to the type of the [parent] Property  
682 component's {name}.

683

684 The following is an XML representation of the ValidValues component:

685

```
686 <ValidValues  
687   {anyAttribute}* >  
688   <documentation />?  
689   {any}*  
690 </ValidValues>
```

691

692 The ValidValues *element information item* has the following Infoset properties:

- 693 • A [local name] of "ValidValues" .
- 694 • A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- 695 • zero or more *attribute information items* amongst its [attributes] as follows:
  - 696 ○ Zero or more namespace qualified *attribute information items*. The [namespace  
697 name] of such *attribute information items* MUST NOT be "http://docs.oasis-  
698 open.org/wsrf/rmd-1".
- 699 • Zero or more *element information items* amongst its [children], either:
  - 700 ○ An OPTIONAL documentation *element information item* (See section 9).
  - 701 ○ Zero or more namespace-qualified *element information items*. The [namespace  
702 name] of such *element information items* MUST NOT be "http://docs.oasis-  
703 open.org/wsrf/rmd-1".

- 704                   ▪ Each *element information item* MUST be an XML fragment that  
705                   corresponds to the type of the XML element identified by the [parent]  
706                   Property component's {name}  
707                   ▪ Note, because these are XML fragments, it is not expected that a  
708                   processor of a MetadataDescriptor document would need to validate  
709                   these element information items (processContents = "skip").  
710                   ○ Zero or more character information items.  
711

### 712 **8.3 ValidValueRange**

713 The ValidValueRange component is an alternative mechanism to specify the set of ValidValues  
714 for the [parent] Property component's {name}. Unlike the ValidValues component, which specifies  
715 an enumeration of values, the ValidValueRange restricts the set of valid values for a {name} by  
716 specifying a range of possible values. This mechanism can only be used when the {name} is an  
717 XML element of simpleType.

718 ValidValueRange defines an optional inclusive lower bound of the range and optional inclusive  
719 upper bound of the range. Both MAY be specified. At least one MUST be specified. The values of  
720 the lower bound and upper bound (if specified) MUST correspond to the value space definition of  
721 the {name}. If the {lowerBound} of this attribute information is NOT specified, its default value is  
722 defined by the lowest possible value defined for the value space of the {name} or "undefined".  
723 Similarly the default value of {upperBound} is the largest value for the value space of the {name}  
724 or "undefined".

725

726 If the {validValues} of a Property component is not empty and contains a ValidValueRange  
727 description, then any Web service that implements the portType or interface identified by  
728 {interface} MUST ensure that the value(s) of the resource property as identified by the {name} of  
729 the Property component MUST correspond to a value within the range specified by {validValues}.

730

731 The properties of a ValidValueRange component are as follows:

- 732                   • {lowerBound} the (inclusive) lower bound of the value space defined by this component  
733                   for the [parent] Property component's {name}.
- 734                   • {upperBound} the (inclusive) upper bound of the value space defined by this component  
735                   for the [parent] Property component's {name}.
- 736                   • {range} a range of values, bounded by the values of {lowerBound} and {upperBound}.  
737                   The values within {range} MUST be compliant with any value space constraints specified  
738                   on the type definition of the [parent] Property component's {name}.

739

740 The following is an XML representation of the ValidValues component:

741

742  
743  
744  
745  
746  
747  
  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778

```
<ValidValueRange
  lowerBound="xs:anySimpleType" ? upperBound="xs:anySimpleType" ?
  {anyAttribute}* >
  <documentation />?
  {any}*
</ValidValueRange>
```

The ValidValueRange *element information item* has the following Infoset properties:

- A [local name] of "ValidValueRange".
- A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- one or more *attribute information items* amongst its [attributes] as follows:
  - one or more attribute information items amongst:
    - An OPTIONAL lowerBound attribute information item
      - The value of this *attribute information item* defines an inclusive lower bound on the range of valid values to apply to the [parent] Property component's {name}.
      - The type of the lowerBound *attribute information item* is an xs:anySimpleType. This type MUST correspond to the type of the [parent] Property component's {name}.
      - The value of this *attribute information item* MUST conform to any value space constraints specified on the type definition of the [parent] Property component's {name}.
    - An OPTIONAL upperBound attribute information item
      - The value of this *attribute information item* defines an inclusive upper bound on the range of valid values to apply to the [parent] Property component's {name}.
      - The type of the upperBound *attribute information item* is an xs:anySimpleType. This type MUST correspond to the type of the [parent] Property component's {name}.
      - The value of this *attribute information item* MUST conform to any value space constraints specified on the type definition of the [parent] Property component's {name}.
  - Zero or more namespace qualified *attribute information items*. The [namespace name] of such *attribute information items* MUST NOT be "http://docs.oasis-open.org/wsrf/rmd-1".
- Zero or more *element information items* amongst its [children], in order as follows:
  - An OPTIONAL documentation *element information item* (See section 9).

- 779                   o Zero or more namespace-qualified *element information items*. The [namespace  
780 name] of such *element information items* MUST NOT be "http://docs.oasis-  
781 open.org/wsrf/rmd-1".  
782

## 783 **8.4 StaticValues**

784 The purpose of the StaticValues component is to define the minimum set of values that a [parent]  
785 Property component's {name} must contain.

786

787 If the {staticValues} of a Property component is not empty, any Web service that implements the  
788 portType or interface identified by {interface} MUST ensure that all the value(s) defined in  
789 {staticValues} appear in the {name}.

790

791 The values contained in a StaticValues component MUST conform to the XML Schema definition  
792 of the {name}. Note: because the child *element information items* of a StaticValues *element*  
793 *information item* are XML fragments, it is not required that these fragments be validated  
794 (processContents is "skip").

795

796 The properties of a StaticValues component are as follows:

- 797     • {values} zero or more XML fragments that correspond to the type of the [parent] Property  
798     component's {name}.
- 799     • The number of XML fragments within {values} MUST NOT be greater than the  
800     maxOccurs facet of the schema declaration target of the {name}.

801

802 The following is an XML representation of the StaticValues component:

803

```
804 <StaticValues  
805     {anyAttribute}* >  
806     <documentation />?  
807     {any}*  
808 </StaticValues>
```

809

810 The StaticValues *element information item* has the following Infoset properties:

- 811     • A [local name] of "StaticValues" .
- 812     • A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- 813     • zero or more *attribute information items* amongst its [attributes] as follows:

- 814                   ○ Zero or more namespace qualified *attribute information items*. The [namespace  
815 name] of such *attribute information items* MUST NOT be "http://docs.oasis-  
816 open.org/wsr/rmd-1".
- 817                   • Zero or more *element information items* amongst its [children], either:
- 818                   ○ An OPTIONAL documentation *element information item* (See section 9).
- 819                   ○ Zero or more namespace-qualified *element information items*. The [namespace  
820 name] of such *element information items* MUST NOT be "http://docs.oasis-  
821 open.org/wsr/rmd-1".
- 822                   ▪ Each *element information item* MUST be an XML fragment that  
823 corresponds to the type of the XML element identified by the [parent]  
824 Property component's {name}
- 825                   ▪ Note, because these are XML fragments, it is not expected that a  
826 processor of a MetadataDescriptor document would need to validate  
827 these element information items (processContents = "skip").  
828

## 829 **8.5 InitialValues**

830 The purpose of the InitialValues component is to define the set of values that a [parent] Property  
831 component's {name} will contain when a WS-Resource becomes available for the first time. If the  
832 {initialValues} of a Property component is not empty, any Web service that implements the  
833 portType or interface identified by {interface} MUST ensure that all the value(s) defined in  
834 {initialValues} appear in the {name} when the service comes online. There is no guarantee as to  
835 how long these values will be present before they are modified; they are different from values  
836 defined in {staticValues} because they are mutable.

837

838 The values contained in a InitialValues component MUST conform to the XML Schema definition  
839 of the {name}. Note: because the child *element information items* of a InitialValues *element*  
840 *information item* are XML fragments, it is not required that these fragments be validated  
841 (processContents is "skip").

842

843 The properties of a InitialValues component are as follows:

- 844                   • {values} zero or more XML fragments that correspond to the type of the [parent] Property  
845 component's {name}.
- 846                   • The number of XML fragments within {values} MUST NOT be greater than the  
847 maxOccurs facet of the schema declaration target of the {name}.

848

849 The following is an XML representation of the InitialValues component:

850

851

852  
853  
854  
855  
856  
  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876

```
<InitialValues  
  {anyAttribute}* >  
  <documentation />?  
  {any}*  
</ InitialValues>
```

The InitialValues *element information item* has the following Infoset properties:

- A [local name] of "InitialValues" .
- A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- zero or more *attribute information items* amongst its [attributes] as follows:
  - Zero or more namespace qualified *attribute information items*. The [namespace name] of such *attribute information items* MUST NOT be "http://docs.oasis-open.org/wsrf/rmd-1".
- Zero or more *element information items* amongst its [children], either:
  - An OPTIONAL documentation *element information item* (See section 9).
  - Zero or more namespace-qualified *element information items*. The [namespace name] of such *element information items* MUST NOT be "http://docs.oasis-open.org/wsrf/rmd-1".
    - Each *element information item* MUST be an XML fragment that corresponds to the type of the XML element identified by the [parent] Property component's {name}
    - Note, because these are XML fragments, it is not expected that a processor of a MetadataDescriptor document would need to validate these element information items (processContents = "skip").

## 877 **9 Documentation Component**

878 The WS-Resource Metadata MetadataDescriptor specification uses the documentation *element*  
879 *information item* as a container for human readable and/or machine processable documentation  
880 in a fashion similar to that defined for WSDL 2.0 [WSDL 2.0]. The content of the *element*  
881 *information item* is "mixed" content as defined in XML Schema [XML Schema]. The  
882 documentation *element information item* may be contained by any *element information item*  
883 defined in this specification.

884

885 The following is an XML representation of the Documentation component:

886

```
887 <documentation {anyAttribute}* >  
888   {any} *  
889 </documentation>
```

890

891 The `documentation element information item` contains the following:

- 892 • A [local name] of "documentation".
- 893 • A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- 894 • Zero or more attribute information items.
  - 895 ○ Zero or more namespace qualified *attribute information items*. The [namespace
  - 896 name] of such *attribute information items* MUST NOT be "http://docs.oasis-
  - 897 open.org/wsrf/rmd-1".
- 898 • Zero or more child *element information items* amongst its [children].

899

## 900 **10 Obtaining a MetadataDescriptor Document**

901 There are two mechanisms that a requestor can use to obtain a WS-Resource  
902 MetadataDescriptor document:

- 903 1. A specific attribute extension to WSDL 1.1 portType definition
- 904 2. A specific Resource Property element.

905

### 906 **10.1 Extending WSDL 1.1 PortType**

907 A WS-Resource MetadataDescriptor document is associated with a WSDL 1.1 portType definition  
908 using an extension of the WSDL 1.1 portType element information item. If any aspect of the  
909 portType is associated with a MetadataDescriptor document, then the portType element MUST  
910 be extended in the manner described below. This extension is described as follows:

911

912

913

914

```
<wsdl:definitions ...>
  <wsdl:portType ...
    wsrm:Descriptor="xs:QName"?
    wsrm:DescriptorLocation="xs:anyURI"?
  ... >
...
</wsdl:portType>
```

918

919

920

921

922 This definition is further constrained as follows:

923 /wsdl:portType/@wsrm:Descriptor

924

If this attribute appears on a WSDL 1.1 portType element its value MUST be a QName that corresponds to a MetadataDescriptor component. Further, the value of the MetadataDescriptor component contained in that document MUST have {interface} that matches the QName of the portType containing @wsrm:Descriptor. Any service that implements this portType MUST be associated with a MetadataDescriptor that is identified by the value of this attribute.

925

926

927

928

929

930

/wsdl:portType/@wsrm:DescriptorLocation

931

If this attribute appears on a WSDL 1.1 portType element its value MUST be a URI. The URI corresponds to a URL at which can be found more information about that MetadataDescriptor namespace, such as an XML document containing a Definitions element as its root element.

932

933

934

935

936

## 10.2 Using Resource Property Elements to expose MetadataDescriptors

937

938

Clients may find and read the wsrm:MetadataDescriptor of a WS-resource using a "metadata WS-resource" that is associated with the resource. The purpose of the metadata WS-resource is to expose a metadata document via its resource properties document. The endpoint reference of the metadata WS-resource is of type wsrm:MetadataDescriptorReference and is exposed in the original WS-resource's resource property document; the form of this resource property is:

939

940

```
<xsd:element name="MetadataDescriptorReference"
  type="wsrm:MetadataDescriptorReferenceType"/>
```

941

942

The constraints on this element are as follows:

943

944

949 /wsrmd:MetadataDescriptorReference

950 This element is an wsa:EndpointReference to a "metadata WS-Resource" associated  
951 with the target WS-Resource. This metadata WS-Resource has a resource properties  
952 document that is equivalent in content to the metadata descriptor document of the target  
953 WS-Resource. The metadata WS-Resource MUST restrict its resource properties  
954 document such that the cardinality of the wsrmd:MetadataDescriptor child elements is  
955 one rather than zero-to-many. This allows for a single wsrmd:MetadataDescriptor to  
956 describe each WS-Resource instance.

957

958

959

## 960 11 References

### 961 11.1 Normative

- 962     **[RFC2119]**     S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,  
963                     <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 964     **[URI]**            T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI):  
965                     Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox  
966                     Corporation, August 1998.
- 967     **[WS-Addressing]**     <http://www.w3.org/2005/08/addressing.pdf>
- 968     **[WS-BaseNotification]**   [http://docs.oasis-open.org/wsn/wsn-ws\\_base\\_notification-1.3-  
969                     spec-pr-02.pdf](http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-pr-02.pdf)
- 970     **[WS-Resource]**       [http://docs.oasis-open.org/wsr/wsrf-ws\\_resource-1.2-spec-pr-  
971                     02.pdf](http://docs.oasis-open.org/wsr/wsrf-ws_resource-1.2-spec-pr-02.pdf)
- 972     **[WS-ResourceProperties]** [http://docs.oasis-open.org/wsr/wsrf-ws\\_resource\\_properties-1.2-  
973                     spec-pr-02.pdf](http://docs.oasis-open.org/wsr/wsrf-ws_resource_properties-1.2-spec-pr-02.pdf)
- 974     **[XML-Infoset]**    *W3C Recommendation "XML Information Set". Available at*  
975                     <http://www.w3.org/TR/xml-infoset/>
- 976     **[XML-Names]**    *W3C Recommendation "Namespaces in XML". Available at*  
977                     <http://www.w3.org/TR/REC-xml-names/>
- 978

### 979 11.2 Non-Normative

- 980     **[WSDL2.0]**       W3C Recommendation "Web Services Description Language" Available at  
981                     <http://www.w3.org/TR/wsdl20/>
- 982
- 983     **[AppNotes]**       [http://www.oasis-  
984                     open.org/apps/org/workgroup/wsr/download.php/16355/wsrf-  
985                     application\\_notes-1.2-notes-pr-02.pdf](http://www.oasis-open.org/apps/org/workgroup/wsr/download.php/16355/wsrf-application_notes-1.2-notes-pr-02.pdf)
- 986

987 **Appendix A. Acknowledgments**

988 The following individuals were members of the committee during the development of this  
989 specification:

990

991 In addition, the following people made contributions to this specification:

992

993

994 **Appendix B. XML Schema for WS-**  
995 **ResourceMetadataDescriptor**

996 The XML types and elements used in this specification are defined in the following XML Schema.  
997

```
998 <?xml version="1.0" encoding="UTF-8"?>  
999 <!--  
1000  
1001  
1002 OASIS takes no position regarding the validity or scope of any  
1003 intellectual property or other rights that might be claimed to pertain  
1004 to the implementation or use of the technology described in this  
1005 document or the extent to which any license under such rights might or  
1006 might not be available; neither does it represent that it has made any  
1007 effort to identify any such rights. Information on OASIS's procedures  
1008 with respect to rights in OASIS specifications can be found at the  
1009 OASIS website. Copies of claims of rights made available for  
1010 publication and any assurances of licenses to be made available, or the  
1011 result of an attempt made to obtain a general license or permission for  
1012 the use of such proprietary rights by implementors or users of this  
1013 specification, can be obtained from the OASIS Executive Director.  
1014  
1015 OASIS invites any interested party to bring to its attention any  
1016 copyrights, patents or patent applications, or other proprietary rights  
1017 which may cover technology that may be required to implement this  
1018 specification. Please address the information to the OASIS Executive  
1019 Director.  
1020  
1021 Copyright (C) OASIS Open (2005-2006). All Rights Reserved.  
1022
```

1023 This document and translations of it may be copied and furnished to  
1024 others, and derivative works that comment on or otherwise explain it or  
1025 assist in its implementation may be prepared, copied, published and  
1026 distributed, in whole or in part, without restriction of any kind,  
1027 provided that the above copyright notice and this paragraph are  
1028 included on all such copies and derivative works. However, this  
1029 document itself may not be modified in any way, such as by removing the  
1030 copyright notice or references to OASIS, except as needed for the  
1031 purpose of developing OASIS specifications, in which case the  
1032 procedures for copyrights defined in the OASIS Intellectual Property  
1033 Rights document must be followed, or as required to translate it into  
1034 languages other than English.

1035  
1036 The limited permissions granted above are perpetual and will not be  
1037 revoked by OASIS or its successors or assigns.

1038  
1039 This document and the information contained herein is provided on an  
1040 "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED,  
1041 INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE  
1042 INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED  
1043 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

1044  
1045  
1046 -->

```
1047 <xsd:schema
1048   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1049   xmlns="http://www.w3.org/2001/XMLSchema"
1050   xmlns:wsa="http://www.w3.org/2005/08/addressing"
1051   xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-2"
1052   targetNamespace="http://docs.oasis-open.org/wsrf/rmd-1"
1053   xmlns:wsrmd="http://docs.oasis-open.org/wsrf/rmd-1"
1054   elementFormDefault="qualified">
1055
1056   <xsd:import
1057     namespace="http://docs.oasis-open.org/wsrf/rp-2"
1058     schemaLocation="http://docs.oasis-open.org/wsrf/rp-2.xsd" />
1059
1060   <xsd:import
1061     namespace="http://www.w3.org/2005/08/addressing"
1062     schemaLocation="http://www.w3.org/2005/08/addressing.xsd" />
1063
1064
1065
```

```

1066
1067 <!-- ===== Utility Types ===== -->
1068 <xsd:simpleType name="PairsOfURIType">
1069   <xsd:list itemType="xsd:anyURI" />
1070 </xsd:simpleType>
1071
1072 <!-- ===== PortType Attribute Extensions ===== -
1073 ->
1074   <xsd:attribute name="Descriptor" type="xsd:QName" />
1075
1076   <xsd:attribute name="DescriptorLocation" type="xsd:anyURI" />
1077
1078 <!-- ===== Documentation Component ===== -->
1079 <xsd:complexType name="DocumentationType" mixed="true" >
1080   <xsd:sequence>
1081     <xsd:any namespace="##any"
1082       minOccurs="0" maxOccurs="unbounded"
1083       processContents="lax" />
1084   </xsd:sequence>
1085   <xsd:anyAttribute/>
1086 </xsd:complexType>
1087
1088 <xsd:complexType name="DocumentedType">
1089   <xsd:sequence>
1090     <xsd:element name="documentation" type="wsrmd:DocumentationType"
1091       minOccurs="0" maxOccurs="1" />
1092   </xsd:sequence>
1093 </xsd:complexType>
1094
1095 <!-- ===== Definitions Component ===== -->
1096 <!--
1097 <Definitions
1098   targetNamespace="xsd:anyURI"
1099   {anyAttribute}* >
1100
1101   <documentation />?
1102   <MetadataDescriptor /> *
1103   {any}*
1104
1105 </Definitions>
1106 -->
1107
1108 <xsd:complexType name="DefinitionsType" >

```

```

1109     <xsd:complexContent>
1110       <xsd:extension base="wsrmd:DocumentedType">
1111         <xsd:sequence>
1112           <xsd:element ref="wsrmd:MetadataDescriptor"
1113             minOccurs="0" maxOccurs="unbounded" />
1114           <xsd:any namespace="##other"
1115             minOccurs="0" maxOccurs="unbounded"
1116             processContents="lax" />
1117         </xsd:sequence>
1118         <xsd:attribute name="targetNamespace"
1119           type="xsd:anyURI" use="required"/>
1120         <xsd:anyAttribute namespace="##other" processContents="lax"/>
1121       </xsd:extension>
1122     </xsd:complexContent>
1123 </xsd:complexType>
1124
1125 <xsd:element name="Definitions" type="wsrmd:DefinitionsType" >
1126   <xsd:key name="MetadataDescriptor">
1127     <xsd:annotation>
1128       <xsd:documentation>
1129         To form a QName, the name of any MetadataDescriptor must be
1130         unique within a Definitions element.
1131       </xsd:documentation>
1132     </xsd:annotation>
1133     <xsd:selector xpath="wsrmd:MetadataDescriptor" />
1134     <xsd:field xpath="@name" />
1135   </xsd:key>
1136 </xsd:element>
1137
1138 <!-- ===== MetadataDescriptor Component ===== -
1139 ->
1140 <!--
1141 <MetadataDescriptor
1142   name="xsd:NCName"
1143   interface="xsd:QName"
1144   wsdlLocation="list of xsd:anyUri"?
1145   {anyAttribute}* >
1146
1147   <documentation />?
1148   <Property /> *
1149   {any}*
1150
1151 </MetadataDescriptor>

```

```

1152  -->
1153
1154  <xsd:complexType name= "MetadataDescriptorType" >
1155    <xsd:complexContent>
1156      <xsd:extension base="wsrmd:DocumentedType">
1157        <xsd:sequence>
1158          <xsd:element ref="wsrmd:Property"
1159            minOccurs="0" maxOccurs="unbounded" />
1160          <xsd:any namespace="##other"
1161            minOccurs="0" maxOccurs="unbounded"
1162            processContents="lax" />
1163        </xsd:sequence>
1164        <xsd:attribute name="name"
1165          type="xsd:NCName" use="required"/>
1166        <xsd:attribute name="interface"
1167          type="xsd:QName" use="required"/>
1168        <xsd:attribute name="wsdlLocation"
1169          type="wsrmd:PairsOfURIType" />
1170        <xsd:anyAttribute namespace="##other" processContents="lax"/>
1171      </xsd:extension>
1172    </xsd:complexContent>
1173  </xsd:complexType>
1174
1175  <xsd:element name="MetadataDescriptor"
1176    type="wsrmd:MetadataDescriptorType" />
1177
1178  <!-- ===== Property Component ===== -->
1179  <!--
1180  <Property
1181    name="xsd:QName"
1182    mutability="[constant|appendable|mutable]" ?
1183    modifiability="[read-only|read-write]" ?
1184    subscribability="xs:boolean" ?
1185    {anyAttribute}* >
1186
1187  <documentation />?
1188  [ <ValidValues> {any}* </ValidValues> |
1189    <ValidValueRange lowerBound='xsd:simpleType'
1190      upperBound='xsd:simpleType' >
1191    </ValidValueRange> ] ?
1192  <StaticValues> {any}* </StaticValues> ?
1193
1194  {any} *

```

```

1195
1196 </Property>
1197 -->
1198 <xsd:complexType name= "PropertyType" >
1199 <xsd:complexContent>
1200 <xsd:extension base="wsrmd:DocumentedType">
1201 <xsd:sequence>
1202 <xsd:choice>
1203 <xsd:element ref="wsrmd:ValidValues"
1204 minOccurs="0" maxOccurs="1" />
1205 <xsd:element ref="wsrmd:ValidValueRange"
1206 minOccurs="0" maxOccurs="1" />
1207 </xsd:choice>
1208 <xsd:element ref="wsrmd:StaticValues"
1209 minOccurs="0" maxOccurs="1" />
1210 <xsd:any namespace="##other"
1211 minOccurs="0" maxOccurs="unbounded"
1212 processContents="lax" />
1213 </xsd:sequence>
1214 <xsd:attribute name="name"
1215 type="xsd:QName" use="required"/>
1216 <xsd:attribute name="mutability"
1217 type="wsrmd:MutabilityType" />
1218 <xsd:attribute name="modifiability"
1219 type="wsrmd:ModifiabilityType" />
1220 <xsd:attribute name="subscribability" type="xsd:boolean"
1221 default="false" />
1222 <xsd:anyAttribute namespace="##other" processContents="lax"/>
1223 </xsd:extension>
1224 </xsd:complexContent>
1225 </xsd:complexType>
1226
1227 <xsd:element name="Property" type="wsrmd:PropertyType" />
1228
1229 <xsd:simpleType name="MutabilityType">
1230 <xsd:restriction base="xsd:string" >
1231 <xsd:enumeration value="constant" />
1232 <xsd:enumeration value="appendable" />
1233 <xsd:enumeration value="mutable" />
1234 </xsd:restriction>
1235 </xsd:simpleType>
1236
1237 <xsd:simpleType name="ModifiabilityType">

```

```

1238     <xsd:restriction base="xsd:string" >
1239         <xsd:enumeration value="read-only" />
1240         <xsd:enumeration value="read-write" />
1241     </xsd:restriction>
1242 </xsd:simpleType>
1243
1244 <!-- ===== Valid Values Component ===== -->
1245 <!--
1246 <ValidValues
1247     {anyAttribute}* >
1248     <documentation />?
1249     {any}*
1250 </ValidValues>
1251 -->
1252 <xsd:complexType name= "ValidValuesType" mixed="true">
1253     <xsd:sequence>
1254         <xsd:element name="documentation" type="wsrmd:DocumentationType"
1255             minOccurs="0" maxOccurs="1" />
1256
1257         <xsd:any namespace="##other"
1258             minOccurs="0" maxOccurs="unbounded"
1259             processContents="lax" />
1260     </xsd:sequence>
1261     <xsd:anyAttribute namespace="##other" processContents="lax"/>
1262 </xsd:complexType>
1263
1264 <xsd:element name="ValidValues" type="wsrmd:ValidValuesType" />
1265
1266 <!-- ===== Valid Range Component ===== -->
1267 <!--
1268 <ValidValueRange
1269     lowerBound="xs:anySimpleType" ? upperBound="xs:anySimpleType" ?
1270     {anyAttribute}* >
1271     <documentation />?
1272     {any}*
1273 </ValidValueRange>
1274 -->
1275 <xsd:complexType name= "ValidValueRangeType" mixed="true">
1276     <xsd:sequence>
1277         <xsd:element name="documentation" type="wsrmd:DocumentationType"
1278             minOccurs="0" maxOccurs="1" />
1279
1280     <xsd:any namespace="##other"

```

```

1281         minOccurs="0" maxOccurs="unbounded"
1282         processContents="lax" />
1283     </xsd:sequence>
1284     <xsd:attribute name="lowerBound" type="xsd:anySimpleType" />
1285     <xsd:attribute name="upperBound" type="xsd:anySimpleType" />
1286     <xsd:anyAttribute namespace="##other" processContents="lax"/>
1287 </xsd:complexType>
1288
1289     <xsd:element name="ValidValueRange" type="wsrmd:ValidValueRangeType"
1290 />
1291
1292 <!-- ===== Static Values Component ===== -->
1293 <!--
1294 <StaticValues
1295     {anyAttribute}* >
1296     <documentation />?
1297     {any}*
1298 </StaticValues>
1299 -->
1300 <xsd:complexType name="StaticValuesType" mixed="true">
1301     <xsd:sequence>
1302         <xsd:element name="documentation" type="wsrmd:DocumentationType"
1303             minOccurs="0" maxOccurs="1" />
1304
1305         <xsd:any namespace="##other"
1306             minOccurs="0" maxOccurs="unbounded"
1307             processContents="lax" />
1308     </xsd:sequence>
1309     <xsd:anyAttribute namespace="##other" processContents="lax"/>
1310 </xsd:complexType>
1311
1312     <xsd:element name="StaticValues" type="wsrmd:StaticValuesType" />
1313
1314 <!-- ===== Initial Values Component ===== -->
1315 <!--
1316 <InitialValues
1317     {anyAttribute}* >
1318     <documentation />?
1319     {any}*
1320 </InitialValues>
1321 -->
1322 <xsd:complexType name="InitialValuesType" mixed="true">
1323     <xsd:sequence>

```

```

1324     <xsd:element name="documentation" type="wsrmd:DocumentationType"
1325                 minOccurs="0" maxOccurs="1" />
1326
1327     <xsd:any namespace="##other"
1328             minOccurs="0" maxOccurs="unbounded"
1329             processContents="lax" />
1330 </xsd:sequence>
1331 <xsd:anyAttribute namespace="##other" processContents="lax"/>
1332 </xsd:complexType>
1333
1334 <xsd:element name="InitialValues" type="wsrmd:InitialValuesType" />
1335
1336
1337 <!-- ===== MetadataDescriptorReference RP GED ===== -->
1338 <xsd:complexType name="MetadataDescriptorReferenceType">
1339   <xsd:complexContent>
1340     <xsd:extension base="wsa:EndpointReferenceType"/>
1341   </xsd:complexContent>
1342 </xsd:complexType>
1343
1344 <xsd:element name="MetadataDescriptorReference"
1345             type="wsrmd:MetadataDescriptorReferenceType" />
1346
1347 <!--
1348
1349 Metadata Resource RP Doc
1350
1351 This defines one property - MetadataDescriptor - which must have a
1352 cardinality of one.
1353
1354 -->
1355
1356 <xsd:element name="MetadataResourceRP" type="wsrmd:DefinitionsType"/>
1357
1358 </xsd:schema>

```

## Appendix C. Revision History

Rev	Date	By Whom	What
wd-01	2004-10-07	Tom Maguire	Initial version created based on work in response to issue 10.
wd-04	2005-10-31	Dan Jemiolo	Updates to original based on TC revisions in summer/fall of 2005.
wd-06	2005-12-12	Dan Jemiolo	Clean up remaining revisions for public review. Re-inserted some features based on requests from WSDM TC.
wd-09	2006-06-04	Dan Jemiolo	Made changes related to MetadataDescriptorReference (an EPR exposed via WSRP that allows a client to read the MDD). Also (re-)added the InitialValues concept to Property.
wd-10	2006-06-19	Dan Jemiolo	Clarified nature of MetadataResourceRP and fixed some example text (T. Banks).
cd-01	2006-06-28	Dan Jemiolo	Changed status to PR.

## 1361 **Appendix D. Notices**

1362 OASIS takes no position regarding the validity or scope of any intellectual property or other rights  
1363 that might be claimed to pertain to the implementation or use of the technology described in this  
1364 document or the extent to which any license under such rights might or might not be available;  
1365 neither does it represent that it has made any effort to identify any such rights. Information on  
1366 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS  
1367 website. Copies of claims of rights made available for publication and any assurances of licenses  
1368 to be made available, or the result of an attempt made to obtain a general license or permission  
1369 for the use of such proprietary rights by implementors or users of this specification, can be  
1370 obtained from the OASIS Executive Director.

1371

1372 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
1373 applications, or other proprietary rights which may cover technology that may be required to  
1374 implement this specification. Please address the information to the OASIS Executive Director.

1375

1376 Copyright (C) OASIS Open (2005). All Rights Reserved.

1377

1378 This document and translations of it may be copied and furnished to others, and derivative works  
1379 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,  
1380 published and distributed, in whole or in part, without restriction of any kind, provided that the  
1381 above copyright notice and this paragraph are included on all such copies and derivative works.  
1382 However, this document itself may not be modified in any way, such as by removing the copyright  
1383 notice or references to OASIS, except as needed for the purpose of developing OASIS  
1384 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
1385 Property Rights document must be followed, or as required to translate it into languages other  
1386 than English.

1387

1388 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
1389 successors or assigns.

1390

1391 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
1392 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO  
1393 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE  
1394 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A  
1395 PARTICULAR PURPOSE.