Web Services Resource Metadata 1.0 (WS-ResourceMetadataDescriptor)

Public Review Draft 01, June 27, 2006

Document identifier:

wsrf-ws_resource_metadata_descriptor-1.0-spec-pr-01

Location:

http://docs.oasis-open.org/wsrf/wsrf-ws_resource_metadata_descriptor-1.0-spec-pr-01.pdf

Artifact Type:

Specification

Technical Committee:

OASIS Web Services Resource Framework TC

Chair(s):

Dr. Ian Robinson, IBM
Dr. David Snelling, Fujitsu Laboratories of Europe

Editors:

Dan Jemiolo, IBM – danjemiolo@us.ibm.com

Abstract:

The components introduced by the WS Resource Framework (WSRF) address functional aspects of modeling stateful resources (such as systems resources) using Web services. WSRF uses WSDL (currently WSDL 1.1) as the form of service description. There is a need to be able to supplement the descriptive information available about a WS-Resource. The format of the information about the components of a WS-Resource is standardized by WSRF, most notably in the resource properties document [WS-ResourceProperties].

In the realm of resource properties, the loosely coupled operations for reading and writing of properties [WS-ResourceProperties] would benefit from metadata. An example of this type of metadata is the mutability constraints and an enumeration of possible values for resource property elements. This document explains the need for such metadata and proposes an information model representing it that would be applicable to Manageable Resources and WS-Resources in general.

Status:

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

This document was approved as a public review draft by the WSRF-TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at www.oasis-open.org/committees/wsrf/.

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

The non-normative errata page, if any, for this specification is located at www.oasis-open.org/committees/wsrf/.

Table of Contents

1 Introduction

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

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

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

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

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

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

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

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

1.1 Goals and Requirements

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

1.1.1 Requirements

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

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

1.1.2 Non-Goals

The following topics are outside the scope of this specification:

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

1.2 Terminology

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

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

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

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

`*′ denotes zero or more occurrences,

`+′ one or more occurrences,

`[′ and `]′ are used to form groups,

`|′ represents choice.

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

```
<!-- sample pseudo-schema -->
<element
      required_attribute_of_type_QName="xs:QName"
      optional_attribute_of_type_string="xs:string"? >
  <required_element />
  <optional_element />?
  <one_or_more_of_these_elements />+
  [ <choice_1 /> | <choice_2 /> ]*
</element>
```

1.3 Namespaces

The following namespaces are used in this document:

Prefix	Namespace
xs	http://www.w3.org/2001/XMLSchema
wsdl	http://schemas.xmlsoap.org/wsdl
wsrf-rp	http://docs.oasis-open.org/wsrf/rp-2
wsrmd	http://docs.oasis-open.org/wsrf/rmd-1

2 Terminology and Concepts

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

Metadata:

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

MetadataDescriptor:

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

MetadataDescriptor Document:

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

3 Example

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

3.1 The OperatingSystem portType

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

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

The *OperatingSystem* portType is sketched as follows:

```
(01)  … xmlns:os=”http://example.com/ns/OperatingSystem”
(02)  … xmlns:id=”http://example.com/ns/Identification”
(03)  …
(04)  <portType name="OperatingSystem"
(05)    wsrf-rp:ResourceProperties="os:OSResourceProperties"
(06)  ..wsrmd:Descriptor=”os:OperatingSystemMetadataDescriptor”
(07)  ..wsrmd:DescriptorLocation=”http://example.com/OperatingSystem.wsrmd” >
(08)  …
(09)  </portType>
```

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

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

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

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

3.2 Operating System Properties

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

3.2.1 Operating System Resource Property definitions

```
(10)  …
(11)    xmlns:os="…
(12)
(13)  <element name="OSResourceProperties">
(14)    <complexType>
(15)      <sequence>
(16)        <element ref="id:ResourceType" minOccurs="0" maxOccurs="1"/>
(17)        <element ref="id:ResourceID" minOccurs="0" maxOccurs="1"/>
(18)        <element ref="os:numberOfProcesses" minOccurs="0" maxOccurs="1"/>
(19)        <element ref="os:totalSwapSpaceSize" minOccurs="0" maxOccurs="1"/>
(20)        <element ref="os:processor" minOccurs="1" maxOccurs="unbounded"/>
(21)      </sequence>
(22)    </complexType>
(23)  </element>
(24)
(25)  <element name="numberOfProcesses" type="xsd:int" />
(26)  <element name="totalSwapSpaceSize" type="xsd:unsignedLong" />
(27)  <element name="processor" type="xsd:string" />
(28)
```

3.2.2 Identification Property definitions

```
(29)  <element name="IdentificationResourceProperties">
(30)    <complexType>
(31)      <sequence>
(32)        <element ref="ResourceType" minOccurs="0" maxOccurs="1"/>
(33)        <element ref="ResourceID" minOccurs="0" maxOccurs="1"/>
(34)      </sequence>
(35)    </complexType>
(36)  </element>
```

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

3.2.3 MetadataDescriptor for Identification portType

An example MetatdataDescriptor document for the Identification portType is included below. For the purposes of this example, this MetadataDescriptor document will be located at the URL `http://example.com/metadataDescriptors/Identification.wsrmd.`

```
(40)
(41)  <Definitions
(42)              xmlns="http://docs.oasis-open.org/wsrf/rmd-1"
(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/wsdl/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>
```

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

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

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

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

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

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

3.2.4 MetadataDescriptor for OperatingSystem portType

For the purposes of this example the MetadataDescriptor document for the OperatingSystem portType is found at `http://example.com metadataDescriptors/OperatingSystem.wsrmd.`

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

```
(58)          <Definitions
(59)              xmlns="http://docs.oasis-open.org/wsrf/rmd-1"
(60)              xmlns:id="http://example.com/ns/Identification"
(61)              xmlns:os="http://example.com/ns/OperatingSystem"
(62)              targetNamespace="http://example.com/ns/OperatingSystem">
(63)            <MetadataDescriptor
(64)              name="OperatingSystemMetadataDescriptor"
(65)              interface="os:OperatingSystem"
(66)              wsdlLocation="http://example.com/ns/OperatingSystem
(67)                            http://example.com/wsdl/OperatingSystem.wsdl">
(68)            <Property name="id:ResourceID"
(69)                      mutability="constant"
(70)                      modifiability="read-only" />
(71)            <Property name="id:ResourceType
(72)                      mutability="constant"
(73)                      modifiability="read-only">
(74)                <ValidValues>
(75)                  <id:ResourceType>SuSELinux</id:ResourceType>
(76)                  <id:ResourceType>IBMzOS</id:ResourceType>
(77)                  <id:ResourceType>MicrosoftWindows_XP</id:ResourceType>
(78)                </ValidValues>
(79)              </Property>
```

```
(80)              <Property name="os:numberOfProcesses"
(81)                        mutability="mutable"
(82)                        modifiability="read-only" />
(83)              <Property name=”os:processor”
(84)                        mutability=”constant”
(85)                        modifiability=”read-only”>
(86)                 <ValidValues>
(87)                   <os:processor>Pentium Family</os:processor>
(88)                   <os:processor>Power PC 750</os:processor>
(89)                   <os:processor>68xxxx Family</os:processor>
(90)                   <os:processor>PA-RISC Family</os:processor>
(91)                   <os:processor>Alpha Family<os:processor>
(92)                   <os:processor>IBM390 Family</os:processor>"
(93)                   <os:processor>G5</os:processor>
(94)                   <os:processor>AMD<os:processor>
(95)                 </ValidValues>
(96)              </Property>
(97)            </MetadataDescriptor>
(98)          </Definitions>
```

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

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

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

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

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

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

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

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

os:numberOfProcesses can not be changed by a requestor using Web services message exchanges such as the SetResourceProperties operation as defined in WS-ResourceProperties [WS-ResourceProperties].

The next Property element references the os:processor. The modifiability and mutability values indicate that the property is static – it will not change during the resource's lifetime. Lines (87)-(94) describe valid values for the os:processor.

4 Logical Model for Metadata

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

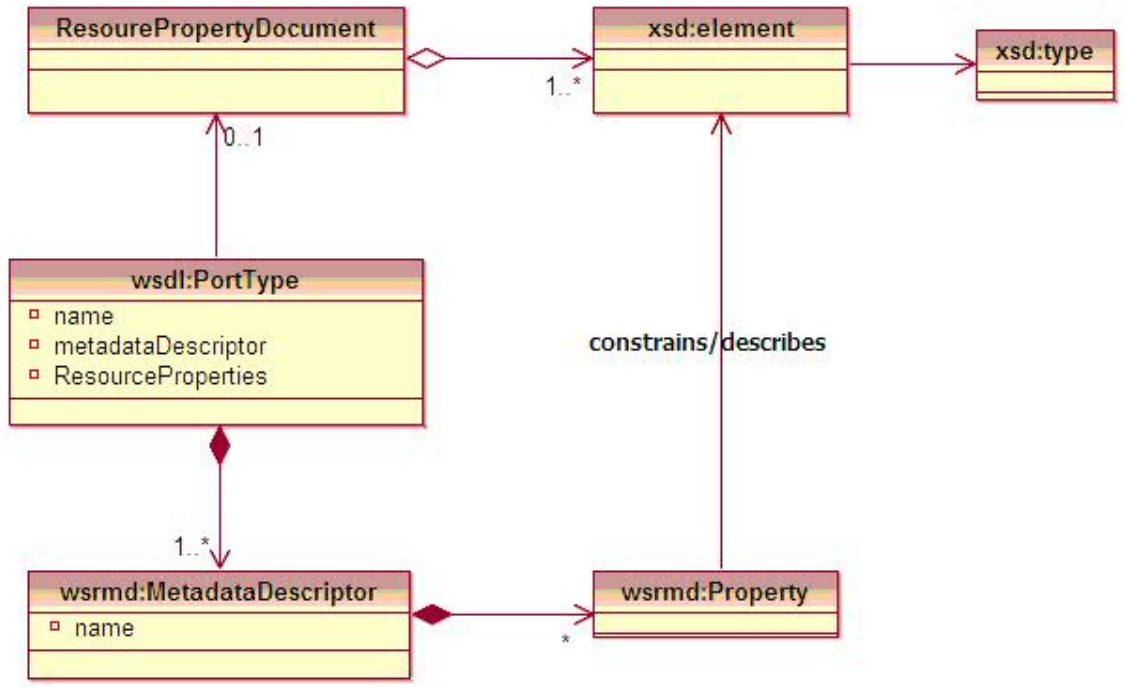

Figure 1 Logical Model of WS-Resource MetadataDescriptor

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

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

5 Information Model for WS-Resource Metadata

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

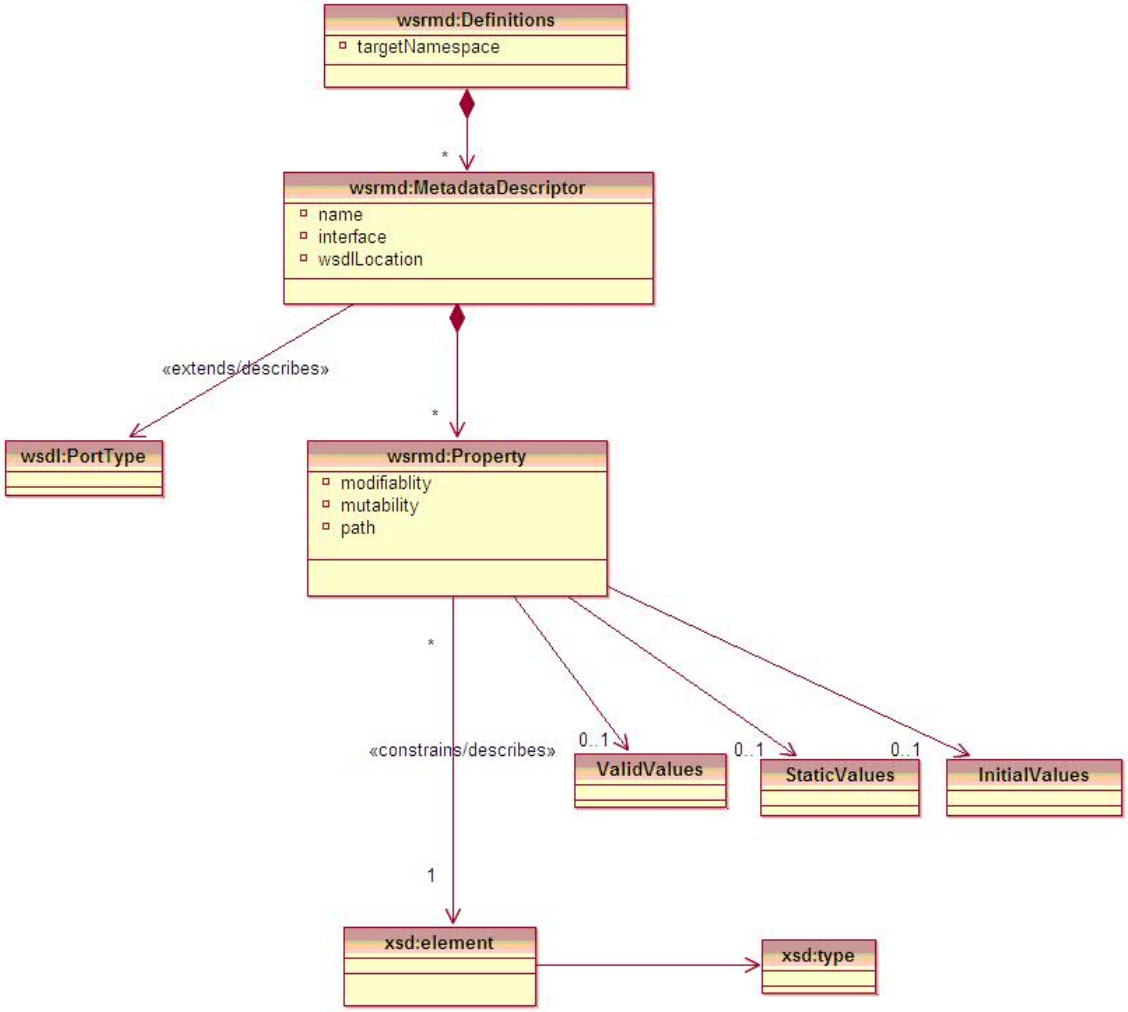

Figure 2 - Information Model for WS-Resource Metadata Descriptor

We describe the Definitions, MetadataDescriptor, and Property components in the following 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).

6.1 MetadataDescriptor components within a Definitions component

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

7 MetadataDescriptor Component

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

The properties of a MetadataDescriptor component are as follows:

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

The following is an XML representation of the MetadataDescriptor component:

```
<MetadataDescriptor
      name="xs:NCName"
      interface="xs:QName"
      wsdlLocation="list of xs:anyUri"?
      {anyAttribute}* >
  <documentation /> ?
  <Property /> *
  {any}*
</MetadataDescriptor>
```

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

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

- two or more *attribute information items* amongst its [attributes] as follows:
  - A REQUIRED name attribute information item
    - The value of this *attribute information item* contains the name of this MetadataDescriptor component.
    - The type of the name *attribute information item* is xs:NCName.
  - A REQUIRED interface attribute information item
    - The value of this *attribute information item* contains a QName of a WSDL 1.1 portType element or WSDL 2.0 interface element associated with this MetadataDescriptor component.
    - The type of the interface *attribute information item* is xs:QName.
  - An OPTIONAL wsdlLocation attribute information item
    - The value of this *attribute information item* contains a list of pairs of URIs; where the first URI of the pair, which MUST be an absolute URI as defined in [URI], indicates a WSDL namespace name, and, the second a hint as to the location of a WSDL document defining WSDL components for that namespace name. The second URI of a pair MAY be absolute or relative.
    - The type of the wsdlLocation *attribute information item* is list of 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 9).
  - Zero or more Property *element information items* (See section 8)
  - 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".

7.1 Properties component of a MetadataDescriptor

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

8 Property Component

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

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

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

```
<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>
```

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

- A [local name] of “Property” .
- 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 name attribute information item
        - The value of this *attribute information item* MUST contain a QName of the Resource Property (an XML Schema global element definition) contained within the Resource Properties document associated with the portType or interface identified by {interface}. The Resource Property element MUST conform to the requirements specified for Resource Property declarations in WS-ResourceProperties.
    - An OPTIONAL mutability attribute information item
        - The value of this *attribute information item* expresses how the value of the {name} can change over time.
        - The type of the mutability *attribute information item* is an xs:string restricted to the following enumeration:
            - “constant”

              The values of the {name} MUST NOT change after WS-Resource creation.
            - “mutable”

              The values of the {name} MAY change at any time during the lifetime of the WS-Resource. Existing values MAY be removed and new values MAY be added.
            - “appendable”

              The values of the {name} MAY have new values added during the lifetime of the WS-Resource. Once added those values MUST NOT be removed.
        - If the mutability *attribute information item* is not defined, the value of the mutability property is “unknown”.
    - An OPTIONAL modifiability *attribute information item*
        - The value of this *attribute information item* indicates whether a requestor can modify the value of the {name}.
        - The type of the modifiability *attribute information item* is an xs:string restricted to the following enumeration:
            - *“read-only”* – The value of the {name} can not be changed by Web services message exchanges such as the SetResourceProperty message as defined in WS-ResourceProperties.
            - *“read-write”* – The value of the {name} MAY be changed by Web services message exchanges such as the SetResourceProperty

message as defined in WS-ResourceProperties. Note - If the value of the modifiability *attribute information item* is “read-write” then the value of the mutability *attribute information item* MUST NOT be “constant”.

- If the modifiability attribute information item is not defined, the value of the modifiability property is “unknown”.

- An OPTIONAL subscribability *attribute information item*
  - The value of this *attribute information item* expresses whether the Resource Property element associated with the {name} can be the target of a subscription. The default value is “false”. Note: The actual subscription semantics are dependent on whatever notification mechanism, if any, (such as WS-BaseNotification [WS-BaseNotification]) is supported.
- 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).
  - An OPTIONAL *element information item* from among the following:
    - A ValidValues element information item (See section 8.2)
    - A ValidValueRange element information item (See section 8.3)
    - An OPTIONAL StaticValues element information item (See section 8.4)
    - An OPTIONAL InitialValues element information item (See section 8.5)
  - 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".

8.1 XML Schema value space and {validValues}

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

8.2 ValidValues

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

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

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

The properties of a ValidValues component are as follows.

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

The following is an XML representation of the ValidValues component:

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

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

- A [local name] of "ValidValues" .
- 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").

  - Zero or more character information items.

8.3 ValidValueRange

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

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

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

The properties of a ValidValueRange component are as follows:

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

The following is an XML representation of the ValidValues component:

```
<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).

- 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".

8.4 StaticValues

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

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

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

The properties of a StaticValues component are as follows:

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

The following is an XML representation of the StaticValues component:

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

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

- A [local name] of "StaticValues" .
- 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").

8.5 InitialValues

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

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

The properties of a InitialValues component are as follows:

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

The following is an XML representation of the InitialValues component:

```
<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”).

9 Documentation Component

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

The following is an XML representation of the Documentation component:

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

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

- A [local name] of “documentation”.
- A [namespace name] of "http://docs.oasis-open.org/wsrf/rmd-1".
- Zero or more attribute information items.
    - 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 child *element information items* amongst its [children].

10 Obtaining a MetadataDescriptor Document

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

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

10.1 Extending WSDL 1.1 PortType

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

```
<wsdl:definitions …>
    <wsdl:portType …
       wsrmd:Descriptor=”xs:QName”?
       wsrmd:DescriptorLocation=”xs:anyURI”?
    … >
…
</wsdl:portType>
```

This definition is further constrained as follows:

/wsdl:portType/@wsrmd:Descriptor

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 @wsrmd:Descriptor. Any service that implements this portType MUST be associated with a MetadataDescriptor that is identified by the value of this attribute.

/wsdl:portType/@wsrmd:DescriptorLocation

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.

10.2 Using Resoure Property Elements to expose MetadataDescriptors

Clients may find and read the wsrmd: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 wsrmd:MetadataDescriptorReference and is exposed in the original WS-resource’s resource property document; the form of this resource property is:

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

The constraints on this element are as follows:

/wsrmd:MetadataDescriptorReference

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

11 References

11.1 Normative

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

[URI] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.

[WS-Addressing] http://www.w3.org/2005/08/addressing.pdf

[WS-BaseNotification] http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-pr-02.pdf

[WS-Resource] http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-pr-02.pdf

[WS-ResourceProperties] http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-pr-02.pdf

[XML-Infoset] *W3C Recommendation "XML Information Set". Available at* http://www.w3.org/TR/xml-infoset/

[XML-Names] *W3C Recommendation "*Namespaces in XML*". Available at* http://www.w3.org/TR/REC-xml-names/

11.2 Non-Normative

[WSDL2.0] W3C Recommendation "Web Services Description Language" Available at http://www.w3.org/TR/wsdl20/

[AppNotes] http://www.oasis-open.org/apps/org/workgroup/wsrf/download.php/16355/wsrf-application_notes-1.2-notes-pr-02.pdf

Appendix A. Acknowledgments

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

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

Appendix B. XML Schema for WS-ResourceMetadataDescriptor

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

```
<?xml version="1.0" encoding="UTF-8"?>
<!--


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

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

Copyright (C) OASIS Open (2005-2006). All Rights Reserved.
```

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

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

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


-->

<xsd:schema
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-2"
  targetNamespace="http://docs.oasis-open.org/wsrf/rmd-1"
  xmlns:wsrmd="http://docs.oasis-open.org/wsrf/rmd-1"
  elementFormDefault="qualified">

  <xsd:import
    namespace="http://docs.oasis-open.org/wsrf/rp-2"
    schemaLocation="http://docs.oasis-open.org/wsrf/rp-2.xsd" />

  <xsd:import
    namespace="http://www.w3.org/2005/08/addressing"
    schemaLocation="http://www.w3.org/2005/08/addressing.xsd" />

```

```
<!-- ======================== Utility Types ======================== -->
  <xsd:simpleType name="PairsOfURIType">
    <xsd:list itemType="xsd:anyURI" />
  </xsd:simpleType>

<!-- ================  PortType Attribute Extensions ================ -
->
  <xsd:attribute name="Descriptor" type="xsd:QName" />

  <xsd:attribute name="DescriptorLocation" type="xsd:anyURI" />

<!-- ================= Documentation Component =================== -->
  <xsd:complexType name="DocumentationType" mixed="true" >
    <xsd:sequence>
      <xsd:any namespace="##any"
           minOccurs="0" maxOccurs="unbounded"
           processContents="lax" />
    </xsd:sequence>
    <xsd:anyAttribute/>
  </xsd:complexType>

  <xsd:complexType name="DocumentedType">
    <xsd:sequence>
      <xsd:element name="documentation" type="wsrmd:DocumentationType"
               minOccurs="0" maxOccurs="1" />
    </xsd:sequence>
  </xsd:complexType>

<!-- ================== Definitions Component ==================== -->
<!--
  <Definitions
      targetNamespace=”xsd:anyURI”
      {anyAttribute}* >

  <documentation />?
  <MetadataDescriptor /> *
  {any}*

</Definitions>
 -->

  <xsd:complexType name= "DefinitionsType" >
```

```
    <xsd:complexContent>
      <xsd:extension base="wsrmd:DocumentedType">
        <xsd:sequence>
          <xsd:element ref="wsrmd:MetadataDescriptor"
                       minOccurs="0" maxOccurs="unbounded" />
          <xsd:any namespace="##other"
               minOccurs="0" maxOccurs="unbounded"
               processContents="lax" />
        </xsd:sequence>
        <xsd:attribute name="targetNamespace"
                       type="xsd:anyURI" use="required"/>
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Definitions" type="wsrmd:DefinitionsType" >
    <xsd:key name="MetadataDescriptor">
      <xsd:annotation>
        <xsd:documentation>
          To form a QName, the name of any MetadataDescriptor must be
          unique within a Definitions element.
        </xsd:documentation>
      </xsd:annotation>
      <xsd:selector xpath="wsrmd:MetadataDescriptor" />
      <xsd:field xpath="@name" />
    </xsd:key>
  </xsd:element>

<!-- =============== MetadataDescriptor Component ================== -
->
<!--
<MetadataDescriptor
      name=”xsd:NCName”
      interface=”xsd:QName”
      wsdlLocation=”list of xsd:anyUri”?
      {anyAttribute}* >

  <documentation />?
  <Property /> *
  {any}*

</MetadataDescriptor>
```

```
-->

  <xsd:complexType name= "MetadataDescriptorType" >
    <xsd:complexContent>
      <xsd:extension base="wsrmd:DocumentedType">
        <xsd:sequence>
          <xsd:element ref="wsrmd:Property"
                       minOccurs="0" maxOccurs="unbounded" />
          <xsd:any namespace="##other"
               minOccurs="0" maxOccurs="unbounded"
               processContents="lax" />
        </xsd:sequence>
        <xsd:attribute name="name"
                       type="xsd:NCName" use="required"/>
        <xsd:attribute name="interface"
                       type="xsd:QName" use="required"/>
        <xsd:attribute name="wsdlLocation"
                       type="wsrmd:PairsOfURIType" />
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="MetadataDescriptor"
               type="wsrmd:MetadataDescriptorType" />

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

  <documentation />?
  [ <ValidValues> {any}* </ValidValues> |
    <ValidValueRange lowerBound=’xsd:simpleType’
                     upperBound=’xsd:simpleType’>
    </ValidValueRange> ] ?
  <StaticValues> {any}* </StaticValues> ?

  {any} *
```

```
</Property>
-->
  <xsd:complexType name= "PropertyType" >
    <xsd:complexContent>
      <xsd:extension base="wsrmd:DocumentedType">
        <xsd:sequence>
          <xsd:choice>
            <xsd:element ref="wsrmd:ValidValues"
                         minOccurs="0" maxOccurs="1" />
            <xsd:element ref="wsrmd:ValidValueRange"
                         minOccurs="0" maxOccurs="1" />
          </xsd:choice>
          <xsd:element ref="wsrmd:StaticValues"
                       minOccurs="0" maxOccurs="1" />
          <xsd:any namespace="##other"
                   minOccurs="0" maxOccurs="unbounded"
                   processContents="lax" />
        </xsd:sequence>
        <xsd:attribute name="name"
                       type="xsd:QName” use="required"/>
        <xsd:attribute name="mutability"
                       type="wsrmd:MutabilityType" />
        <xsd:attribute name="modifiability"
                       type="wsrmd:ModifiabilityType" />
        <xsd:attribute name="subscribability" type="xsd:boolean"
default="false" />
        <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Property" type="wsrmd:PropertyType" />

  <xsd:simpleType name="MutabilityType">
    <xsd:restriction base="xsd:string" >
      <xsd:enumeration value="constant" />
      <xsd:enumeration value="appendable" />
      <xsd:enumeration value="mutable" />
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="ModifiabilityType">
```

```
    <xsd:restriction base="xsd:string" >
      <xsd:enumeration value="read-only" />
      <xsd:enumeration value="read-write" />
    </xsd:restriction>
  </xsd:simpleType>

<!-- ================ Valid Values Component ==================== -->
<!--
<ValidValues
      {anyAttribute}* >
  <documentation />?
  {any}*
</ValidValues>
-->
  <xsd:complexType name= "ValidValuesType" mixed="true">
    <xsd:sequence>
      <xsd:element name="documentation" type="wsrmd:DocumentationType"
                   minOccurs="0" maxOccurs="1" />

      <xsd:any namespace="##other"
               minOccurs="0" maxOccurs="unbounded"
               processContents="lax" />
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:element name="ValidValues" type="wsrmd:ValidValuesType" />

<!-- ================ Valid Range Component ==================== -->
<!--
<ValidValueRange
      lowerBound="xs:anySimpleType" ? upperBound="xs:anySimpleType" ?
      {anyAttribute}* >
  <documentation />?
  {any}*
</ValidValueRange>
-->
  <xsd:complexType name= "ValidValueRangeType" mixed="true">
    <xsd:sequence>
      <xsd:element name="documentation" type="wsrmd:DocumentationType"
                   minOccurs="0" maxOccurs="1" />

      <xsd:any namespace="##other"
```

```
                minOccurs="0" maxOccurs="unbounded"
                processContents="lax" />
    </xsd:sequence>
    <xsd:attribute name="lowerBound" type="xsd:anySimpleType" />
    <xsd:attribute name="upperBound" type="xsd:anySimpleType" />
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:element name="ValidValueRange" type="wsrmd:ValidValueRangeType"
/>

<!-- ================ Static Values Component ==================== -->
<!--
<StaticValues
      {anyAttribute}* >
  <documentation />?
  {any}*
</StaticValues>
-->
  <xsd:complexType name= "StaticValuesType" mixed="true">
    <xsd:sequence>
      <xsd:element name="documentation" type="wsrmd:DocumentationType"
                   minOccurs="0" maxOccurs="1" />

      <xsd:any namespace="##other"
                minOccurs="0" maxOccurs="unbounded"
                processContents="lax" />
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:element name="StaticValues" type="wsrmd:StaticValuesType" />

<!-- ================ Initial Values Component ================== -->
<!--
<InitialValues
      {anyAttribute}* >
  <documentation />?
  {any}*
</InitialValues>
-->
  <xsd:complexType name= "InitialValuesType" mixed="true">
    <xsd:sequence>
```

```
      <xsd:element name="documentation" type="wsrmd:DocumentationType"
                   minOccurs="0" maxOccurs="1" />

      <xsd:any namespace="##other"
               minOccurs="0" maxOccurs="unbounded"
               processContents="lax" />
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:element name="InitialValues" type="wsrmd:InitialValuesType" />


<!-- =========== MetadataDescriptorReference RP GED =============== -->
  <xsd:complexType name="MetadataDescriptorReferenceType">
    <xsd:complexContent>
      <xsd:extension base="wsa:EndpointReferenceType"/>
    </xsd:complexContent>
  </xsd:complexType>

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

<!--

Metadata Resource RP Doc

This defines one property – MetadataDescriptor – which must have a
cardinality of one.

-->

  <xsd:element name=”MetadataResourceRP” type=”wsrmd:DefinitionsType”/>

</xsd:schema>
```

1367 # 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.

1368

Appendix D. Notices

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

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