Web Services Distributed Management: Management Using Web Services (MUWS 1.1) Part 2

OASIS Standard, 01 August 2006

Document identifier:

wsdm-muws2-1.1-spec-os-01

Location:

http://docs.oasis-open.org/wsdm/wsdm-muws2-1.1-spec-os-01.pdf

Technical Committee:

OASIS Web Services Distributed Management TC

Chair(s):

Heather Kreger, IBM, <*kreger@us.ibm.com*>

Editor:

Vaughn Bullard, AmberPoint, Inc. <*vbullard@amberpoint.com>*

William Vambenepe, Hewlett-Packard <vbp@hp.com>

Abstract:

There are two specifications produced by the Web services Distributed Management technical committee: Management *Using* Web services (MUWS) and Management *Of* Web services (MOWS, see **[MOWS]**). This document is part of MUWS.

MUWS defines how an Information Technology resource connected to a network provides manageability interfaces such that the IT resource can be managed locally or from remote locations using Web services technologies.

MUWS is composed of two parts. This document is MUWS part 2 and provides specific messaging formats used to enable the interoperability of MUWS implementations. MUWS part 1 **[MUWS Part 1]** provides the fundamental concepts for management using Web services. MUWS part 2 depends on MUWS part 1 while part 1 is independent of part 2.

Status:

This document was last revised or approved by the membership of OASIS 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. This document is updated periodically on no particular schedule.

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/wsdm/.

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/wsdm/ipr.php.

The non-normative errata page for this specification is located at www.oasis-open.org/committees/wsdm.

Table of Contents

1 Introduction

This document, MUWS Part 2, builds upon the foundation provided by **[MUWS Part 1]**. All of the normative text presented in MUWS Part 1 is considered normative text for MUWS Part 2. All informational text presented in MUWS Part 1 is relevant informational text for MUWS Part 2. Compliance with MUWS Part 1 is REQUIRED for every aspect of MUWS Part 2.

The text of this specification along with Appendix C (Schemas), Appendix D (WSDL Elements), Appendix E (Topics) and Appendix F (Description of Situation Types) is considered normative with the following exceptions: the abstract, the examples and any section explicitly marked as non-normative.

The terminology and notational conventions defined in **[MUWS Part 1]** apply to this document.

The following namespaces are used, unless specified otherwise.

Prefix	Namespace
muws1	http://docs.oasis-open.org/wsdm/muws1-2.xsd
muws2	http://docs.oasis-open.org/wsdm/muws2-2.xsd
muwsw	http://docs.oasis-open.org/wsdm/muws-2.wsdl
muwse	http://docs.oasis-open.org/wsdm/muwse-2.xml
wsnt	http://docs.oasis-open.org/wsn/b-2.xsd
wstop	http://docs.oasis-open.org/wsn/t-1
wsrf-rp	http://docs.oasis-open.org/wsrf/rp-2
wssg	http://docs.oasis-open.org/wsrf/sgw-2.wsdl
wsdl	http://www.w3.org/2002/07/wsdl
wsa	http://www.w3.org/TR/ws-addr-core
soap	http://schemas.xmlsoap.org/soap/envelope/ or http://www.w3.org/2002/12/soap-envelope
xs	http://www.w3.org/2001/XMLSchema

XML elements (**[XML1.0 3rd Edition]**]) and schema (**[XML Schema Part 1]** and **[XML Schema Part 2]**) types introduced in this section belong to the namespace mapped to “muws2”.

WSDL (**[WSDL]**) elements introduced in this section belong to the namespace mapped to “muws-2.wsdl”.

2 Use of the Web Services Platform

As a complement to the Web services platform described in **[MUWS Part 1]**, MUWS Part 2 presents an additional set of specifications in order to achieve interoperability among disparate implementations of MUWS. This goal is achieved by the precise specification of the format for each management message.

2.1 Use of WS-Addressing and the WS-Resource concept

MUWS Part 2 depends upon concepts presented in the Web Services Resources Framework (WSRF). A manageable resource is a refinement of a WSRF resource. A WS-Resource is defined as the actual composition of a resource and a web service from which the resource can be accessed. In addition, a reference to a manageability endpoint relies upon reference mechanisms as defined in the WSRF WS-Resource Specification **[WS-Resource]**, and more specifically, leverages and refines the endpoint reference concept, as defined in WS-Addressing.

The MUWS specification does not currently define how to obtain an EPR. Currently, to obtain an EPR, there may be some out-of-band agreement between a service provider and a manageability consumer. Possibly, some future version of the MUWS specification might clarify and standardize an approach to obtain an EPR. This specification provides some guidelines on discovering EPRs for manageability endpoints and the WSDM Primers explain how to enable and use discovery.

In the specific case where a manageability endpoint corresponds to one and only one manageable resource, then either the endpoint reference concept defined in WS-Addressing can be used. If no endpoint reference exists, then the consumer can use a URL to identify the manageable resource. Following this reasoning, a manageability consumer without an endpoint reference for a manageable resource MAY try to invoke manageability operations without including an endpoint reference. If such an invocation succeeds, the manageability consumer can infer it is accessing a manageable resource through a manageability provider.

2.2 Use of WS-Resource Properties

Management properties as defined in MUWS are represented as WSRF properties, and use the mechanisms defined in WS-ResourceProperties (**[WS-RP]**). In other words, each manageable resource exposes a resource properties document containing, as children of the document root, all the properties of the manageable resource. The manageable resource then makes this document available, as described in WS-ResourceProperties.

Supporting WS-ResourceProperties means that any implementation of an interface that includes properties MUST include access methods to these properties as defined by WS-ResourceProperties. Specifically, the interface MUST include the GetResourceProperty operation defined by **[WS-RP]** and MAY include the GetMultipleProperties, SetResourceProperties, GetResourcePropertiesDocument and QueryResourceProperties operations. If the QueryResourceProperties operation is provided, then the QueryResourceProperties operation SHOULD support the XPath 1.0 query expression dialect, represented by URI http://www.w3.org/TR/1999/REC-xpath-19991116.

2.3 Use of WS-Notification

MUWS uses the notification mechanism described by WS-BaseNotification (**[WSN]**). If a manageability capability includes an ability to offer events to a consumer, then the definition of the capability SHALL include topic space, as described in WS-Topics (**[WST]**). The topic space MUST contain an appropriate set of topics for the events offered by the capability. As described in MUWS Part 1, an event is defined by a topic QName and a content element. The topic is mapped to the topic of the event, as defined by **[WST]**.

As specified by WS-BaseNotification, whether the event payload (of type *muws1:ManagementEvent*) is the first child of the SOAP (**[MOWS]** Kirk Wilson, *Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.1*, OASIS Committee Draft, February 2006, http://docs.oasis-open.org/wsdm/wsdm-mows-1.1-spec-os-01.pdf

[SOAP]) body or whether it is wrapped in a *wsnt:Notify* element is determined based on whether the *wsnt:UseNotify* element in the subscription message is set to *true* or *false*.

Note that WS-BaseNotification does not currently support a means to specify that only some of the information contained in the notification message should be sent to the consumer. MUWS does not define a means to specify this either. The manageability consumer and the implementer of a manageability endpoint should be aware that there is a performance cost for processing many, large notification messages.

2.4 Metadata

MUWS defines a set of base schema for metadata elements. These metadata elements can be represented as XML Schema elements. The purpose of a metadata element is to supplement the information available in the WSDL **[WSDL]** and the WS-ResourceProperties **[WS-RP]** declaration for a manageability interface. A metadata element provides additional description relevant to the managed resource. In particular, a metadata element enables a tool or management application, to perform detailed reasoning and make specialized inferences about a manageable resource at runtime, and, during development, when no instance is available for a manageable resource.

If metadata is required, then an XML document containing metadata is defined and associated with a WS-ResourceProperties document and WSDL. Document processing, such as an XPath query, is used to extract all or part of the metadata. Currently, WSDM does not define the format of, how to associate, or, how to access document metadata content. Although some mechanism is necessary, this MUWS specification does not provide any mechanism for accessing metadata from an instance of a manageable resource.

This MUWS specification does not provide any description of how metadata is associated with a type of manageable resource, stored, or made available.

The MUWS specification defines a set of metadata elements that apply to the basic manageability of a manageable resource. The MUWS specification uses Global Element Declarations to represent a metadata element.

2.4.1 Metadata applicable to all aspects of manageability capabilities

MUWS defines metadata elements applicable to all aspects of a manageability capability (operations, properties, events…). These elements are:

```
<muws2:Capability>xs:anyURI</muws2:Capability> *
```

muws2:Capability metadata element SHOULD be provided for any MUWS aspect of a manageability interface. This enables discovery of aspects of an interface associated with a capability.This URI element identifies which capability an aspect is associated with.

This metadata element indicates the classification of an aspect of an interface according to an intended capability, or capabilities. For example, an aspect may be classified as a metric, or, as a configuration property. A property may be relevant to more than one capability. For example, a configuration property of a computer system contains the IP address but this same property could also be used for identification purposes.

Some of the known capabilities are listed below for illustration. This is not an exhaustive list. For a detailed explanation, see the relevant MUWS manageability capability specification. Additional capabilities are expected to be added as extensions to MUWS.

- *http://docs.oasis-open.org/wsdm/muws/capabilities/Identity*
  Identity capability. See **[MUWS Part 1]**.
- *http://docs.oasis-open.org/wsdm/muws/capabilities/Configuation*
  Configuration property. See section 3.5.
- *http://docs.oasis-open.org/wsdm/muws/capabilities/CorrelatableProperties*
  ”Correlatable Properties” capability. See **[MUWS Part 1]**.
- *http://docs.oasis-open.org/wsdm/muws/capabilities/State*
  State capability. See section 3.1.3.
- *http://docs.oasis-open.org/wsdm/muws/capabilities/Metrics*
  Metrics capability. See section 3.4.
- *User defined*
  A user defined capability that extends, or, is different from, a standard capability defined in MUWS.

```
<muws2:ValidWhile Dialect=”xs:anyURI”> {any} * </muws2:ValidWhile>
```

muws2:ValidWhile contains a statement that, when true, asserts that the interface aspect to which this metadata element is related is valid. This is used, for example, to express the fact that an operation can only be invoked when certain properties have certain values.

muws2:ValidWhile/@muws2:Dialect identifies how the statement in *muws2:ValidWhile* is built and what rules govern its evaluation. MUWS defines one possible value for this element. Other values can also be defined.

The value defined by MUWS is http://www.w3.org/TR/1999/REC-xpath-19991116. When this dialect is used, the content of *muws2:ValidWhile* is an **[XPath 1.0]** expression. This expression is evaluated against the resource properties document of the manageable resource. If the XPath expression evaluates to a Boolean value of *true*, or if it evaluates to a non-empty non-boolean value without any errors, then the statement is considered true.

2.4.2 Metadata applicable to properties

General purpose metadata that is not management specific is defined in the MUWS specification, but not specified in schema. General purpose metadata that can be defined for any property include:

- *Mutability* – indicates if the property value can change over time
- *Modifiability* – indicates if the property can be set directly (not as a side-effect)
- *Valid Values* –a set of valid values for the property
- *Valid Range* – a range of valid values for the property
- *Static Values* – a set of permanent values for the property
- *Notifiability* – indicates if a notification is sent when there is a change to the value of the property

Schema to represent general purpose metadata should be composed from a metadata specification, for example, the WS-Resource Metadata Descriptor **[WSRMD]**, as developed in the WS-RF OASIS technical committee. (***Special note*: Although the WSRF Technical committee is currently working on the WSRMD, the WSDM committee will probably reference the WSRMD in a future release of the WSDM specification.)

In addition, MUWS defines a set of metadata related to management. Any property element may have the following manageability metadata element:

```
<muws2:Units>xs:string</muws2:Units>
```

muws2:Units indicates the default unit for this property as a string.

Other metadata elements, applicable for metric-type properties, are defined in section 3.4.3.

2.4.3 Operations

General purpose metadata, that is not management specific, is defined in the MUWS specification, but not specified in schema. General purpose metadata that can be defined for any operation includes:

- *Idempotency* – indicates if invoking the operation twice is equivalent to invoking it once

Schema to represent general purpose metadata should be composed from a metadata specification, for example, the WS-Resource Metadata Descriptor **[WSRMD]**, as developed in the WS-RF OASIS technical committee.

In addition, MUWS defines metadata related to management. Any operation element may have the following manageability metadata element:

```
<muws2:PostCondition Dialect=”xs:anyURI”>
  {any} *
</muws2:PostCondition>
```

muws2:PostCondition contains a statement that asserts "true" immediately after the corresponding operation is complete.

muws2:PostCondition/@muws2:Dialect is a URI identifying how the statement in *muws2:PostCondition* is built, and what rules govern its evaluation. MUWS defines one possible value for this element. Other values can be defined.

The value defined by MUWS is http://www.w3.org/TR/1999/REC-xpath-19991116. When this dialect is used, the content of *muws2:PostCondition* is an **[XPath 1.0]** expression. This expression is evaluated against the resource properties document of the manageable resource. If the XPath expression evaluates to a Boolean value of *true,* or, if it evaluates to a non-empty non-boolean value without any errors, then the statement is considered true.

2.5 Events

2.5.1 Event Format

[MUWS Part 1] defines the *muws1:ManagementEvent* Global Element Declaration as a container for management events. *muws1:ManagementEvent* allows information to be added via extensibility elements. The *muws2:Situation* element defined below MUST be present as a child of the *muws1:ManagementEvent* element in notifications.

As a result, the event format is flexible and extensible. At the same time, automated analysis is possible, as the event format provides a means to classify an event into one of a limited set of classifications and sub-classifications.

MUWS event classifications are based on a thorough analysis of event types, as produced by a wide range of IT equipment, and grouped according to the general nature of events. For example, virtually all manageable resources have a means of being started. However, almost all managed resources express a start event in some unique way. The basic knowledge that the resource has started is all that is necessary, even for fairly sophisticated, automated management.

2.5.1.1 Situation Element

To support event classifications, the MUWS specification defines the *SituationCategoryType* element, a specialization of a *muws2:CategoryType*. MUWS defines the top level of classifications. Extensions to these classifications enable a refined event classification. Through the use of the extensible *muws2:CategoryType* mechanism, WSDM event consumers can comprehend the situation for an event to a degree commensurate with their ability.

```
<muws2:Situation>
  <muws2:SituationCategory>
    muws2:SituationCategoryType
  </muws2:SituationCategory>
  <muws2:SuccessDisposition>
    (Successful|Unsuccessful)
  </muws2:SuccessDisposition> ?
  <muws2:SituationTime>xs:dateTime</muws2:SituationTime> ?
  <muws2:Priority>xs:short</muws2:Priority> ?
  <muws2:Severity>xs:short</muws2:Severity> ?
  <muws2:Message>muws:LangString</muws2:Message> ?
  <muws2:SubstitutableMsg MsgId=”xs:string” MsgIdType=”xs:anyURI”>
      <muws2:Value>xs:anySimpleType</muws2:Value>*
  </muws2:SubstitutableMsg> ?
</muws2:Situation>
```

muws2:Situation/muws2:SituationCategory categorizes the type of the situation that caused the event report. The values, listed below, represent the names of elements in the *muws2* namespace. The categories are listed in the order of precedence. In a case where there may be some ambiguity about which category to use, the higher precedent category SHOULD be used. The ordering of situation categories is based on empirical data showing relative importance of various types of events. The use of a higher precedent category permits more effective and timely correlation and analysis of events that may indicate the presence of a serious problem. Details and examples for use of the following values are documented in Appendix F. This element is REQUIRED.

- AvailabilitySituation
- CapabilitySituation
- ConfigureSituation
- StopSituation
    - **StopInitiated**
    - **AbortInitiated**
    - **PauseInitated**
    - **StopCompleted**
- StartSituation
    - **StartInitiated**
    - **RestartInitiated**
    - **StartCompleted**
- RequestSituation
    - **RequestInitiated**
    - **RequestCompleted**
- DestroySituation
- CreateSituation
- DependencySituation
- ConnectSituation
- ReportSituation
    - **Performance**
    - **Security**
    - **Heartbeat**
    - **Status**
    - **Trace**
    - **Debug**
    - **Log**
- OtherSituation

muws2:Situation/muws2:SuccessDisposition in the case where this situation is triggered by a command, this value specifies a successful disposition of the command causing a report of this situation. This element is OPTIONAL and should not be included if the situation is not the result of a command. The element is a restriction of the type xs:string allowing the following values:

- `Successful`
- `Unsuccessful`

muws2:Situation/muws2:SituationTime represents the date and time an event is observed. If the value does not include a time zone designation, or, if the value does not use 'Z' for UCT, then the value MUST be interpreted as having a time zone of UCT. The value of SituationTime MUST provide granularity as precise as supported by the generating platform. This is a REQUIRED element and MUST be provided by the component acting as the originator of an event.

muws2:Situation/muws2:Priority represents the importance of an event. This element supports management functions requiring an event to be associated with a priority. This is an OPTIONAL element. Values are constrained to a range from 0 through 100. The predefined priorities are:

- Low (10)
- Medium (50)
- High (70).

Other priorities MAY be used but MUST NOT be less than 0 or greater than 100.

muws2:Situation/muws2:Severity represents the perceived severity of the status the event is describing with respect to the application that reports the event. This element supports management functions requiring an event to be associated with a severity. This is an OPTIONAL element. Severity levels, based upon the DMTF CIM Alert Indications Perceived Severity, are as follows:

- 6 (Fatal): a condition is unrecoverable and the service is no longer available.
- 5 (Critical): a condition affecting the service has occurred. Immediate corrective action is required.
- 4 (Major): a problem of relatively high severity has occurred. It is likely that normal use of the service is impeded.
- 3 (Minor): a problem of relatively low severity has occurred. It is unlikely that normal use of the service is impeded.
- 2 (Warning): a problem affecting the service may occur. Diagnostic and corrective action is recommended.
- 1 (Information): a message output considered as normal and expected. For example, a process begins, a process finishes, or status information is displayed.
- 0 (Unknown): a severity level cannot be determined.

muws2:Situation/muws2:Message represents the text accompanying an event. This is typically the resolved message string in a human-readable format, as rendered for a specific locale, and is of type *muws2:LangString* which is an extension of *xs:string* requiring the *xml:lang* attribute. This is an OPTIONAL property. While the string length for *Message* is unbounded, it is RECOMMENDED that the string length for *Message* does not exceed 1024 characters.

muws2:Situation/muws2:SubstitutableMsg – represents the message data in a substitutable form. The attributes *MsgId* and *MsgIdType* indentify the base message type and text. The element value contains the data that will be formatted according to the formatting rules defined by the *MsgId*. This is an OPTIONAL element. However, if this element is used, it must contain all the attributes and elements specified below.

muws2:Situation/muws2:SubstitutableMsg/@muws2:MsgId specifies the message identifier of an event. This identifier SHOULD be a unique value string, consisting of alphanumeric or numeric characters. The value can be as simple as a string of numeric characters that identify a message in a message catalog. As an alternative, the value can be a multipart string of alphanumeric characters, for example, DBT1234E. This is a REQUIRED attribute. The maximum string length for *MsgId* MUST NOT exceed 256 characters. The *MsgIdType* attribute indicates the formatting type of the *MsgId*.

muws2:Situation/muws2:SubstitutableMsg/@muws2:MsgIdType specifies the meaning and format of the *MsgId*. This is a REQUIRED attribute. The type of the *MsgIdType* attribute is a URI.

muws2:Situation/muws2:SubstitutableMsg/muws2:Value can be of any simple type. There are one or more occurrences of this element with each occurrence containing an *xsi:type* attribute defining the type of the contained data. This element is used to pass data values that are substituted as a message is formatted. This element is OPTIONAL. A *MsgId* and *MsgIdType* define rules to map parameters into a composed message, based upon the order of the *Value* elements.

As an example, a minimal SituationType report for the initiation of a requested restart (at 6:06PM in Greenwich on Nov 11, 2004) would be as follows.

```
<muws2:Situation>
  <muws2:SituationCategory>
    <foo:RestartInitiated>
      <muws2:StartSituation/>
    </foo:RestartInitiated>
  </muws2:SituationCategory>
  <muws2:SuccessDisposition>Succesful</muws2:SuccessDisposition>
  <muws2:SituationTime>2004-11-11T18:06:00Z
  </muws2:SituationTime>
  <muws2:Message xml:lang=”en”>
    Managed Thing XXX: restart processing begun
  </muws2:Message>
</muws2:Situation>
```

Please note, as outlined in the description of *muws2:CategoryType*, the most general situation classification appears as the innermost element within the XML nest.

2.5.1.2 Sequence Number

The sequenceNumber attribute allows multiple messages to be sent and processed in a logical order that could, owing to idiosyncrasies in the delivery, be different than the order in which they arrived at the consumer. The sequence number helps consumers to sort messages into a meaningful flow, as defined by the provider. The `sequenceNumber` is typically used only by event producers when the granularity of the event time stamp (the `reportTime` attribute) is not sufficient to properly sequence events. In other words, the `sequenceNumber` field is typically used to sequence events that have the same time stamp value.

It is REQUIRED that sequence numbers for succeeding events be monotonically increasing throughout at least a single value of timestamp.

This attribute is OPTIONAL. There is no default value.

```
<muws1:ManagementEvent ...
       muws2:sequenceNumber=”xsd:ulong”? />
```

2.5.1.3 Event Correlation Properties

The optional EventCorrelationProperties element allows for the more efficient interpretation and lifecycle management of events. This element is OPTIONAL.

muws1:ManagementEvent/muws1:EventCorrelationProperties: The `EventCorrelationProperties` element for event correlation follows with the EventCorrelationPropertiesType subelements. These properties generally allow for more efficient interpretation and lifecycle management of events. This element is OPTIONAL.

muws1:ManagementEvent/muws1:EventCorrelationProperties/repeatCount: The `repeatCount` specifies the number of occurrences of identical events within a specified time interval. The time interval is specified by the `elapsedTime` property described next. The definition of “identical events” is application-specific and therefore is not defined by this specification.

This property is OPTIONAL and mutable. The `repeatCount` MAY be set by the component that reports the event or by the event consumer. There is no default value. A value of zero or no value indicates no repeated occurrences of the event.

muws1:ManagementEvent/muws1:EventCorrelationProperties/elapsedTime: The `elapsedTime` is the time interval during which a certain number of identical events occurred. The number of occurrences is specified by the value of `repeatCount`. The `elapsedTime` value indicates the duration of time within which the repeated events were observed.

The value of this property MUST be expressed in microsecond granularity.

This property is OPTIONAL and mutable; however, if the `repeatCount` is specified, then an elapsed time MUST be present. The `elapsedTime` MUST be set by the same component that sets the `repeatCount`. There is no default value for `elapsedTime`.

```
<muws1:EventCorrelationProperties …>
  <muws1:repeatCount>xs:short</muws1:repeatCount> ?
  <muws1:elapsedTime>xs:long</muws1:elapsedTime> ?
</muws1:EventCorrelationProperties> ?
```

2.5.1.4 Message Catalog for Internationalization

Late binding of event message text is enabled via *muws2:Situation/muws2:SubstitutableMsg* element described above but the ability to localize message text to a language compatible with the consumer of events may require the ability to reference message catalogs appropriate to the manageable resource reporting the event. The *msgCatalogInformation* element described below provides this additional information.

The *msgCatalogInformation* element provides a means to define message catalog information that is to be used for internationalization of messages contained within the event. These properties enable the optional use of a message catalog for specifying messages.

This is an OPTIONAL property.

The elements within a *msgCatalogInformation* element are:

muws1:ManagementEvent/muws1:msgCatalogInformation/msgCatalog: The *msgCatalog* element is the qualified URI of the message catalog that contains the locale-dependent message template that is indexed by the *muws-p2-xs:Situation/SubstitutableMsg@MsgId* attribute. The format of the messages in *msgCatalog* is specified by the *msgCatalogType* element.

This element is REQUIRED.

muws1:ManagementEvent/muws1:msgCatalogInformation/msgCatalogType: The *msgCatalogType* URI property specifies the format of the *msgCatalog*. The format defines the substitution identifier syntax for the *muws-p2-xs:Situation/SubstitutableMsg/Value* (that is, the method used to insert runtime information contained in the *muws-p2-xs:Situation/SubstitutableMsg/Value* element into the message template that is retrieved from the message catalog to form a completely translated message). The reserved keywords for *msgCatalogType* are:

- Java- The message catalog uses Java properties encoding. See http://java.sun.com/j2se/1.4.2/docs/api/java/util/Properties.html
- XPG- The message catalog uses X/Open XPG specifications for providing internationalization support. See http://www.unet.univie.ac.at/aix/aixprggd/genprogc/nls.htm

Other values may be used for other catalog types.

This element is OPTIONAL. Default value is nil and assumed to be understood by the consumer of the event.

The schema for *msgCatalogInformation* is:

```
<muws2:MsgCatalogInformation>
        <muws2:msgCatalog>xs:anyURI</muws2:msgCatalog>
        <muws2:msgCatalogType>xs:anyURI</muws2:msgCatalogType> ?
</muws2:MsgCatalogInformation>
```

2.5.2 Topics for capabilities

For each capability defined by MUWS, topics are defined that encompass every event related to that capability. For example, if a property related to capability “foo” changes, then a notification is sent to subscribers of the topic corresponding to a change event on this property, as described by **[WS-RP]**. Concurrently, since this property is associated with the “foo” capability, a notification is also sent to subscribers of the topic encompassing change events associated with capability “foo”.

Appendix E contains the XML description of all the topics defined in the MUWS specification. The sections of this document that define a capability also define the topic(s) associated with that capability. The following MUWS topics encompass every event associated with the capability defined in MUWS Part 1:

The *muwse:IdentityCapability* topic defined below is used for events related to the *Identity* capability.

```
<wstop:Topic name="IdentityCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

The *muwse:ManageabilityCharacteristicsCapability* topic defined below is used for events related to the *ManageabilityCharacteristics* capability.

```
<wstop:Topic name="ManageabilityCharacteristicsCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

The *muwse:CorrelatablePropertiesCapability* topic defined below is used for events related to the *CorrelatableProperties* capability.

```
<wstop:Topic name="CorrelatablePropertiesCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

2.6 Representation of Categorization Taxonomies in XML

In the description of several manageability capabilities, categories of information are organized in taxonomies. This is for example the case for the categories of relationships between manageable resources, for operational states of resources, etc. In order to convey category information, including taxonomy lineage, to a manageability consumer, and, in order to represent XML information instances, the following convention is used:

MUWS defines an XML Schema complex type called CategoryType. The content of XML elements of this type is any XML element. When an element is defined of this type, it MUST obey the following rules:

- The element and each descendant has, at most, one child element.

- The top-level element and each descendant represent one category in a taxonomy.
- The top level element represents the most specialized category. Each element represents a more specialized category than the category represented by the element it contains, if any.

The CategoryType XML Schema type is declared as follows:

```
<xs:complexType name=”CategoryType”>
<xs:sequence>
  <xs:any namespace=”##any” minOccurs="0" processContents=”lax”/>
</xs:sequence>
</xs:complexType>
```

The CategoryType type is used to declare an XML element containing instances of general, or unqualified, category information. The CategoryType type is also used to derive an XML Schema type representing a specific category, for example, a relationship among resources, or among operational states.

Category information MUST be declared as follows:

- An XML element declaring which QName identifies the semantics of the category.
- The XML element declaring an XML Schema type which is a restriction of *muws2:Category*, or a specialized XML Schema type derived from some other refinement of *muws2:Category*, for example, *muws2:RelationshipType*.
- The contents of the XML element MUST be either:
  - The one XML element corresponding to the generalization of the currently declared category
  - The empty sequence. This case occurs if the declared category does not have any generalizations. For example, the declared category might be the top of a taxonomy.

For example, assume that information about a maintenance state is represented, using the approach described above. In this example, “off-for-maintenance” is a substate of “offline”, which is a substate of a resource being “unavailable”. The XML representation for this example follows:

```
<mydomain:Off-for-Maintenance>
  <mydomain:Offline>
    <anyresource:Unavailable/>
  </mydomain:Offline>
</mydomain:Off-for-Maintenance>
```

By processing the XML information, a manageability consumer may learn that a resource is in a state identified by the *mydomain:Off-for-Maintenance* element. However, at the same time, if the manageability consumer is not aware of definitions and semantics associated with the *mydomain* namespace, the consumer may safely assume the resource is in the commonly known state identified by *anyresource:Unavailable*. Since the most specialized elements are first encountered, a consumer can generally stop processing an element of type *muws2:Category* as soon as it reaches an element the semantic of which it understands.

3 Capabilities applicable to manageable resources

This section defines capabilities applicable to manageable resources. The capabilities defined in this section complement the capabilities defined in MUWS Part 1.

3.1 Description

The manageability capability URI for the description capability is http://docs.oasis-open.org/wsdm/muws/capabilities/Description

3.1.1 Definition

A manageable resource has a capability description. This description capability gives the manageable resource additional metadata about its capabilities. These attributes include zero or more caption attributes, zero or more description attributes as well as an optional version attribute.

3.1.2 Properties

This capability defines the following properties:

```
<muws2:Caption>muws2:LangString</muws2:Caption> *
```

muws2:Caption contains a descriptive name for the manageable resource. The *Caption* property is intended for human consumption. A *Caption* is expected to be short and is suitable for display next to a graphic icon. *Caption* is a read-write, optional property with a cardinality of 0 to many. *Caption* is of type *muws2:LangType*, which is a restriction of *xs:string* carrying an *xml:lang* attribute. This attribute contains a language identifier as defined by **[RFC3066]**. There can not be more than one *Caption* per language identifier.

Metadata for *Caption*:

It is *Mutable*
It is *Modifiable*
It has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/Description
</muws2:Capability>
```

```
<muws2:Description>muws2:LangString</muws2:Description> *
```

muws2:Description is a string containing a description for the resource being managed. The *Description* property is intended for human consumption. A *Description* is expected to be longer and more detailed than a *Caption*. *Description* is a read-write optional property with a cardinality of 0 to many. *Description* is of type *muws2:LangType*, which is a restriction of *xs:string* carrying an *xml:lang* attribute. This attribute contains a language identifier as defined by **[RFC3066]**. There cannot be more than one *Description* per language identifier.

Metadata for *Description*:

It is *Mutable*
It is *Modifiable*
It has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/Description
</muws2:Capability>
```

```
<muws2:Version>xs:string</muws2:Version> ?
```

muws2:Version is a string representing the version of the resource being managed. MUWS does not specify how this string is constructed. The *Version* string can be specified by any domain-specific specification that uses MUWS. *Version* is an optional property with a cardinality of 0 to1.

Metadata for *Version*:

It is *Mutable*
It is *Modifiable*
It has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/Description
</muws2:Capability>
```

3.1.3 Events

The *muwse:DescriptionCapability* topic defined below is used for events related to the *Description* capability.

```
<wstop:Topic name="DescriptionCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

3.2 State

The manageability capability URI for the State capability is
http://docs.oasis-open.org/wsdm/muws/capabilities/State

3.2.1 Definition

A resource may exhibit behavior according to one or more state models. Since a single definition of an operational state model is not sufficient for all types of resource, the *State* capability is a means to allow different state models to be used by different resources. The state capability provides a pattern for representing any type of state or state model that a manageable resource can expose. This section uses operational state as an example to illustrate the application of this pattern to a simple state model.

Although MUWS defines no state model, there should be a very limited and well defined set of states to facilitate interoperability. Each state is identified by a URI. This URI is exposed by a resource via some resource property.

This capability does not define any specific property, operation or event. A manageability endpoint is said to provide this capability if at least one property exposes state information and follows the pattern described in section 3.2.3.2.

3.2.2 Describing State Models

Each state in a state-machine has a well-defined meaning. It is possible to reuse state definitions in different state machines. States are identified by an element with a particular QName, using the taxonomy scheme defined in section 2.6.

States in the state model may have duration. Transitions between states are considered to be instantaneous.

States can have sub-states that MUST be wholly contained within a higher-level state.

A state model may also define an operation that can be used to affect some transition in the model. Note that a transition may also occur as a result of some internal or external event on the resource.

Each state machine has an associated resource property element exposing a read-only view of the current state of the state machine. Therefore, a consumer cannot change a resource state by modifying a state resource property.

There may be more than one possible transition between two states in the state model. The individual transitions between states are identified by a URI. This identification allows, for example, a receiver of state transition notifications to discern which transition occurred.

Figure 1 shows a simple state model that is used as an example in this section – it does not constitute the specification of a recommended state model.

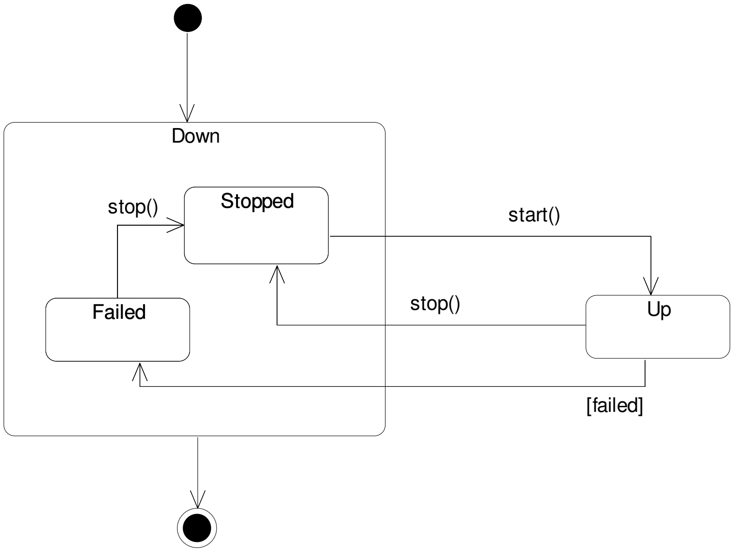

Figure 1: Example Operational State Model

In this example, the state machine is identified by URI http://example.com/StateModels/SimpleOperationalState, bound to namespace prefix *exns*.

In this example, the state model has four states. Each state is represented by elements with a QName, as follows:

- *exns:Down*

  This QName corresponds to the “Down” state in the UML diagram. A resource in this state is unable to perform any of its functional tasks.

- *exns:Stopped*

  This QName corresponds to the “Stopped” sub-state of the “Down” state in the UML diagram. Since this state is a sub-state of the “Down” state, it follows that a resource in the "Stopped" sub-state is unable to perform any of its functional tasks. A manageable resource exposing this state model can be started from the "Stopped" sub-state.

- *exns:Failed*

  This QName corresponds to the “Failed” sub-state of the Down state in the UML diagram. Since this state is a sub-state of the “Down” state, it follows that a resource in the "Failed" sub-state is unable to perform any of its functional tasks. A manageable resource exposing this state model can not be started directly from the “Failed” sub-state. Such a resource must first transition to the “Stopped” sub-state.

- *exns:Up*
  This QName corresponds to the “Up” state in the UML diagram. A resource in this state is able to perform at least some of its functional tasks.

3.2.3 Information Markup Declarations

3.2.3.1 Representation of States

A state, as represented in a state model, may be a top level state or a state that is nested within another state according to some defined taxonomy. MUWS defines a way to represent a state category and its taxonomy lineage, but an actual definition of any category is specific to a particular resource management model. Therefore MUWS defines no state model. In other words, MUWS specifies only the mechanism used to convey a state category in XML. The MUWS mechanism applied to the representation of states is defined as follows:

muws2:StateType XML Schema type is declared as follows

```
<xs:complexType name=”StateType”>
  <xs:complexContent>
    <xs:extension base=”muws2:CategoryType”/>
  </xs:complexContent>
</xs:complexType>
```

The *muws2:StateType* type is used to declare an XML element containing an instance of state.

A state MUST be declared as follows:

- An XML element declaring which QName identifies the semantics of the state.
- The XML element has an XML Schema type of *muws2:StateType*,, or a restriction of *muws2:StateType*.
- The contents of the XML element MUST be either:
  - The one XML element that corresponds to the state containing this state. In other words, this state is a sub-state of another state.
  - The empty sequence. This case occurs if this state is not a sub-state of another state.

For example, the “Failed” state in the example above is a sub-state of the “Down” state. An instance of the “Failed” state may be represented, using the rules described above, by the following XML fragment:

```
<my:StateTypeInstanceElement xsi:type=”StateType”>
  <exns:Failed>
    <exns:Down/>
  </exns:Failed>
</my:StateTypeInstanceElement>
```

3.2.3.2 Representation of state

MUWS defines the following Global Element Declaration (GED) to represent an instance of a state:

```
<muws2:State>muws2:StateType</muws2:State>
```

The State element provides a representation of the state of a manageable resource. The State element follows the convention for the *muws2:CategoryType* type described in section 2.6. This convention allows the rendering of a hierarchy of states and sub-states. State values are defined in the operational state model for the resource. This specification does not define the operational state model for any resource.

3.2.3.3 Representation of state transition

MUWS defines the following Global Element Declaration (GED) which contains an XML representation of a change of state in a state model.

```
<muws2:StateTransition Time“xs:dateTime”
                            TransitionIdentifier=” xs:anyURI”?>
  <muws2:EnteredState>muws2:StateType</muws2:EnteredState>
  <muws2:PreviousState>muws2:StateType</muws2:PreviousState>?
  {any} *
</muws2:StateTransition>
```

muws2:StateTransition is used for representing information about a state change.

muws2:StateTransition/@muws2:Time attribute indicates the time at which the transition occurred (transitions are assumed to be instantaneous). This attribute is REQUIRED.

muws2:StateTransition/@muws2:TransitionIdentifier attribute indicates the actual transition that occurred. This attribute is OPTIONAL and may be omitted where, for example, there is only one transition between the *EnteredState* and the *PreviousState*.

muws2:StateTransition/muws2:EnteredState element indicates which state has been entered during the transition. This element is REQUIRED.

muws2:StateTransition/muws2:PreviousState element indicates the state that the resource was in immediately prior to the state change occurring. This element is OPTIONAL to allow for the time between the state model being created in some initial state, for example when the resource is created, and the time of the transition from that initial state.

3.2.4 Properties

This capability does not define any standard property.

A capability defining a state model SHOULD define a resource property that exposes the state. It is RECOMMENDED that a state model also define a resource property that exposes the last state transition.

The property used to expose the state must either contain the *muws2:State* element or be of type *muws2:StateType*. The name of the property can be any name meaningful to the state model defined in the capability. There may be multiple state capabilities, and therefore multiple state properties for a resource. The metadata for this property SHOULD include the possible values. That is, the state model should provide a list of states in the state model.

The property to represent the last transition, if such a property is provided, must contain the element *muws2:StateTransition*. The name of the last transition property can be any name meaningful to the state model. There may be multiple state capabilities and multiple properties exposing the last transition.

3.2.4.1 Example

Examples of resource properties for an operational state capability could be specified as follows:

```
<foo:OperationalState>
  <muws2:State>...</muws2:State>
</foo:OperationalState>
<foo:LastOperationalStateTransition>
  <muws2:StateTransition>...</muws2:StateTransition>
</foo:LastOperationalStateTransition>?
```

The following fragment provides an example from a resource properties instance document containing the properties defined in this example:

```
<foo:OperationalState>
  <muws2:State>
    <exns:Failed><exns:Down/></exns:Failed>
  </muws2:State>
</foo:OperationalState>
<foo:LastOperationalStateTransition>
  <muws2:StateTransition Time=”2004-03-11T11:30:56Z”
TransitionIdentifier=”http://example.com/SimpleOperationalState/T/Failed”>
    <muws2:EnteredState>
      <exns:Failed><exns:Down/></exns:Failed>
    </muws2:EnteredState>
    <muws2:PreviousState>
      <exns:Up/>
    </muws2:PreviousState>
  </muws2:StateTransition>
</foo:LastOperationalStateTransition>
```

In this example, the *foo:OperationalState* property contains the current operational state of the resource, using the muws2:State element defined in section 3.2.3.2 The *foo:LastOperationalStateTransition* property contains a description of the most recent operational state transition for the resource, using the *muws2:StateTransition* element as defined in section 3.2.3.2.

3.2.5 Operations

A capability defining a state model usually defines any operations that can be used to cause some of the transitions within the state model. These operations are specific to the resource and its state model.

3.2.6 Events

The *muwse:StateCapability* topic defined below is used for events related to the *State* capability.

```
<wstop:Topic name="StateCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

It is RECOMMENDED that resources send a notification on a transition between states. The topic defined for the *State* capability SHALL be used to publish such notifications. If a resource sends such a notification, then the notification message MUST contain at least the XML element representing a state transition (*muws2:StateTransition*).

To obtain events about a certain state transition, a subscriber can use a *Selector,* on the notification subscription, to select only those events containing the required *muws2:TransitionIdentifier* element in the notification content, or, a combination of *muws2:EnteredState* and *muws2:PreviousState* elements in the notification content. The *Selector* mechanism is described in **[WSN]**.

To filter for events about entry into a particular state or set of states, a Selector expression based on the *muws2:EnteredState* element can be used. To filter for events about exit from a particular state or set of states a *Selector* expression based on the *muws2:PreviousState* element can be used.

3.3 Operational Status

The manageability capability URI for this capability is
http://docs.oasis-open.org/wsdm/muws/capabilities/OperationalStatus

3.3.1 Definition

The operational status capability defines a simple representation of the availability of a resource. This is expressed in terms defined by MUWS. These terms are independent of any specific state model, as defined by domain experts. An operational status property reflects whether the resource is available, unavailable, or degraded. Operational status does not conform to a specific state model. Rather, each value may correspond to more than one state in the operational state model, and conversely more than one operational status value may correspond to a single state in the operational state model. The manageable resource provides the appropriate mapping from state to status and sets the *OperationalStatus* property accordingly.

3.3.2 Properties

The operational status properties and elements are specified as follows:

```
<muws2:OperationalStatus>
  (Available|PartiallyAvailable|Unavailable|Unknown)
</muws2:OperationalStatus>
```

The following fragment provides an example from a resource properties instance document containing this property:

```
<muws2:OperationalStatus>Available</muws2:OperationalStatus>
```

The *muws2:OperationalStatus* property is of type *muws2:OperationStatusType*. The type is a restriction of *xs:string* and provides a simple indication of the availability of the resource, independent of the potentially complex operational state model. This property has a cardinality of 1. The valid values are:

- *Available*: This value indicates that a manageable resource is operating normally within any configured operating parameters, and is able to perform all functional tasks.
- *PartiallyAvailable*: This value indicates that a manageable resource is operating, but outside of configured operating parameters. A manageable resource reporting this operational status is able to perform some, but not all, functional tasks. A manageable resource may, for example, be in the process of starting or a resource may be lacking some resource it needs to perform.
- *Unavailable*: This value indicates that a manageable resource is not operating, and is not able to perform any functional tasks. A manageable resource may have been stopped, or may have failed.
- *Unknown*: This value indicates that a manageable resource is unable to report status at this time.

Metadata for *OperationalStatus*:

It is *Mutable*
It is not *Modifiable*
It has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/OperationalStatus
</muws2:Capability>
```

3.3.3 Events

The *muwse:OperationalStatusCapability* topic defined below is used for events related to the *Operational Status* capability.

```
<wstop:Topic name="OperationalStatusCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

No specific event is defined, since the notification on property value change provided by WS-ResourceProperties is sufficient, when applied to the *muws2:OperationalStatus* property.

3.4 Metrics

The manageability capability URI for this capability is http://docs.oasis-open.org/wsdm/muws/capabilities/Metrics

3.4.1 Definition

A metric is a specific type of property. A metric represents a collected value during a collection period. A common characteristic of metrics is that they change over time. This section defines how to represent metrics and the metadata necessary to correctly process and interpret a metric value.

As a simple example, to clarify what a metric is, consider a toll bridge with two properties, the length of the bridge and the number of cars that have passed over the bridge. The length of the bridge, while numeric is not a metric. Length represents a current configuration of the bridge. One can not reset the length of the bridge. By contrast, the number of cars that have passed over the bridge is a metric. It requires collecting, counting, or measuring the number of cars. Typically, a count occurs for some interval, or duration of time, such as the last hour, the last day, or, since the bridge was constructed. One might reset the number of cars, for example, at the start of a new interval.

The following sections define two (2) types of "metadata" used for metrics. The *value modifiers* provides context to the metric's value, improving the consumer's ability to interpret the value.

The *definitional metadata* describes the means by which the value of the metric is acquired.

It is important to keep the two distinct types of metadata or data modifiers in mind when reading the following sections as it is sometimes the case that there are two similar definitions, distinguished primarily by the entity (value or metric) that they describe.

Note: Future revisions of this specification are likely to move this definitional metadata into an external metadata component, since it is used to define the configuration of the metric properties. However, the value modifiers will remain associated with the metric itself, as their values is directly pertinent to the interpretation of the metric value.

3.4.2 Information Markup Declarations

The following schema fragment declares the (reusable) data type used to expose the metrics of a resource. All attributes defined in the *muws2:MetricAttributes* attribute group are OPTIONAL. The MetricAttributes group defines the value modifiers associated with the metric.

```
<xs:attributeGroup name="MetricAttributes">
       <xs:attribute name="ResetAt" type="xs:dateTime"/>
       <xs:attribute name="LastUpdated" type="xs:dateTime"/>
       <xs:attribute name="Duration" type="xs:duration"/>
</xs:attributeGroup>
```

(MetricAttributes) attribute group MUST be included in every metric type or metric type property element declaration.

(MetricAttributes)/ResetAt indicates the time when a metric value was reset. See the definition of *muws2:TimeScope* for information on when to provide this attribute If the attribute value does not include a time zone indication, or Z for UTC, then the value MUST be interpreted as UTC.

(MetricAttributes)/LastUpdated indicates the last update time of a metric value. If the value does not include a time zone indication, or Z for UTC, then the value MUST be interpreted as UTC.

(MetricAttributes)/**Duration** indicates the measurement period relative to the value of the *LastUpdated* metric attribute. This is the time over which a metric value was collected, counted, or measured.

The Duration attribute is valid only for a metric that covers a time interval. Therefore, the *Duration* attribute MUST be included for a metric having a *TimeScope* of *Interval*. Similarly, it MUST NOT be included for a metric having a *TimeScope* of *PointInTime* or *SinceReset*, because these *TimeScope*s imply their measurement period. For these *TimeScope*s, an implementer should make use of *ResetTime* and *CurrentTime* to calculate the duration for the collection of a metric value.

The following metric type definition is an example of how a metric attribute is incorporated into a metric type. All metric types MUST incorporate the *muws2:MetricAttributes* attribute group.

```
<xs:complexType name="MyExampleIntegerMetricType">
  <xs:simpleContent>
    <xs:extension base="xs:integer">
      <xs:attributeGroup ref="muws2:MetricAttributes"/>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

The following fragment shows an example instance of the above metric type.

```
<MyIntegerMetric
    LastUpdated=”2004-03-11T11:30:56Z”
    Duration=”PT1H”>
  12345
</MyIntegerMetric>
```

3.4.3 Metadata

The following metadata is applicable to any property that is a metric:

It is *Mutable*
It is not *Modifiable*
It has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/Metrics
</muws2:Capability>
```

In addition to the value modifiers, the following additional metadata items are defined for a property that is a metric. As opposed to the value modifiers described above, this metadata describes the metric itself, independent of the value at any particular time.

```
<muws2:ChangeType>(Counter|Gauge|Unknown)</muws2:ChangeType>
```

muws2:ChangeType is an enumeration indicating how a change to an associated metric value should be interpreted by a consumer. A property representing a metric MUST include a single instance of *ChangeType* in its metadata description. Each *ChangeType* value is interpreted as follows:

- *Counter* - the value of the metric is a monotonically increasing integer. Such a metric value increases by increments of “1” when a situational event occurs to the resource.
- *Gauge* – changes of the value of the metric are not constrained in the way changes to *Counter* metrics are constrained.
- *Unknown* - the change behavior for the value of the metric is not known or cannot be described.

```
<muws2:TimeScope>
  (Interval|PointInTime|SinceReset)
</muws2:TimeScope>
```

muws2:TimeScope is an enumeration for indicating if there is some interval, over which the data is collected, counted, or measured. A property that is a metric MUST include a single instance of *TimeScope* in its metadata description. Each *TimeScope* value is interpreted as follows:

- *Interval* - the value of a metric is collected over some time interval. In this case a *Duration* attribute MUST be reported with a metric property. The value of a *Duration* attribute is the elapsed time, from the beginning of an interval, to the end of an interval. A *Duration* MAY change, but usually remains the same for every collection of a metric. The *ResetAt* attribute MAY also be reported with such a metric property and may have a value during the measurement period.
- *PointInTime* - the value of a metric is counted, collected, or measured at a single instant in time. In this case a *Duration* attribute MUST NOT be reported with a metric property. A metric defined with a TimeScope of PointInTime does not support a reset capability and MUST NOT include a *ResetAt* attribute.
- *SinceReset* - the value of the metric is collected since the last reset of a resource, or since the manageable resource started collecting data for a metric. . In this case a *Duration* attribute MUST NOT be reported with a metric property, and a *ResetAt* attribute MUST be reported.

```
<muws2:GatheringTime>
  (OnChange|Periodic|OnDemand|Unknown)
</muws2:GatheringTime>
```

muws2:GatheringTime is an enumeration indicating under which circumstance the value of a metric is updated. A property that is a metric MUST include a single instance of *muws2:GatheringTime* in its metadata description. Each *muws2:GatheringTime* value is interpreted as follows:

- *OnChange* - the value of a metric is updated whenever a change occurs to the quantity measured.
- *Periodic* - the value of a metric is updated on a regularly scheduled basis.
- *OnDemand* - the value of a metric is updated when processing a request for the metric value.
- *Unknown* - it is unknown when the value of a metric is updated.

```
<muws2:CalculationInterval>xs:duration</muws2:CalculationInterval>
```

muws2:CalculationInterval represents the interval at which a value of a metric is gathered or calculated by a resource. The value of a metric is not updated during a calculation interval. Unlike *Duration*, which can change every time the metric is updated, the value of *CalculationInterval* is expected to change rarely. This is because *CalculationInterval* is used only for a value of a metric that is updated at regular intervals.

Note also that it is possible for a *CalculationInterval* to be different than the *Duration*. The former specifies the frequency of update while the latter specifies the period over which the data has been collected.

```
<muws2:MetricGroup>xs:anyURI</muws2:MetricGroup>
```

muws2:MetricGroup indicates that a metric property is a member of a group of metrics. A metric property MAY be a member of zero or more metric groups. A metric group is identified by a URI. Each metric property included in a metric group MUST have a *muws2:MetricGroup* element containing an identical URI. A metric property MAY include zero or more *muws2:MetricGroup* elements in its metadata description. Each *muws2:MetricGroup* element represents a membership of the metric property in a metric group.

3.4.3.1 An Example

To illustrate how the value modifiers and definitional metadata might work together, consider a metric that computes average request size over the previous hour. Such a metric might have the following description. (Note that this example uses human readable rather than XML data representations.)

Definitional Metadata	
ChangeType	Gauge
TimeScope	Interval
GatheringTime	Periodic
CalculationInterval	1 Minute
Value Modifiers	
ResetAt	2:30pm
Duration	1 Hour
LastUpdated	4:00pm
(value)	4800

These values describe our average request size that is a gauge that is updated once a minute to produce a sliding scale one hour in length. This instance shows an average size of 4800 from 3-4pm.

3.4.4 Properties

The following fragment provides the specification of a resource metrics property:

```
<muws2:CurrentTime>xs:dateTime</muws2:CurrentTime>
```

muws2:CurrentTime contains the current time, as known to a resource, when a property was retrieved from a manageable resource. This property is useful to a manageability consumer in the absence of a time synchronization mechanism when analyzing the time values received from a manageability endpoint. muws2*:CurrentTime* is a read-only mandatory property with a resource cardinality of 1.

The Metrics capability requires the *muws2:CurrentTime* property to be present in a resource property. The *muws2:CurrentTime* property provides a reference point for time-based attributes, as defined by metric data types. Note that *muws2:CurrentTime* is not a metric. Rather, it is a property of type *xs:dateTime* defined as part of the “Metrics” capability, consequently, any reset operations has no effect on *muws2:CurrentTime*.

3.4.5 Events

The *muwse:MetricsCapability* topic defined below is used for events related to the *Metrics* capability.

```
<wstop:Topic name="MetricsCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

WS-ResourceProperties specifies the ability to define optional topics for a resource property that can emit notifications when a value changes. These topics allow a consumer to request notifications on an update of a metric property.

3.5 Configuration

The manageability capability URI for this capability is
http://docs.oasis-open.org/wsdm/muws/capabilities/Configuration

3.5.1 Definition

A configuration property is any resource property exposing a value that, when changed, changes some operational behavior of the resource.

The value of a configuration property may be changed directly by a set operation, or, may be changed as a side effect of some other operation.

3.5.2 Properties

MUWS does not define any required property for the *Configuration* capability. Domain experts can define configuration properties which are then marked as associated with the configuration capability. The metadata for a configuration property MUST be:

It is *Mutable*

It is *Modifiable* only if the WS-ResourceProperties *SetResourceProperty* operation can be used to change the value of the property. It is not *Modifiable* if the property is changed only as a side effect.

It has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/Configuration
</muws2:Capability>
```

3.5.3 Operations

WS- ResourceProperties *SetResourceProperty* operation MAY be used to change a configuration value.

3.5.4 Events

The *muwse:ConfigurationCapability* topic defined below is used for events related to the *Configuration* capability.

```
<wstop:Topic name="ConfigurationCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

4 Capabilities applicable to management in general

Section 3, "Capabilities applicable to manageable resources", when merged with the capabilities defined in **[MUWS Part 1]**, provide the list of manageability capabilities defined by MUWS. This section provides management-related capabilities that are different from manageability capabilities.

A *manageability capability* is offered by a manageability representation and a manageability capability applies to a resource as represented by a manageability representation. In contrast, a management-*related capability* can be offered by any endpoint of a Web service, not just a manageability endpoint.

The function of a management-related capability is related to the management of a resource, but it is not necessarily offered directly by a manageability endpoint of a resource. For example, the capability to help a manageability consumer discover a new manageable resource can be provided by a registry instead of by a management representation of the resource. As another example, a manageable resource may provide information about relationships in which it participates. The information about a relationship may also provide valid information for another entity or resource that is not manageable, like a registry, maintaining and providing relationship information about a resource without the resource providing the relationship information directly.

4.1 Relationships

The manageability capability URI for this capability is http://docs.oasis-open.org/wsdm/muws/capabilities/Relationships

4.1.1 Definition

A relationship is an N-ary association between resources. A relationship may have properties, operations and other characteristics. One of these properties is a type that conveys the semantic of the relationship. The resources involved in the relationship are called participants. Each participant has a role in the relationship. The participants may or may not be manageable resources in the MUWS sense. The notion of "direction" of a relationship is a semantic interpretation based on role definitions. There could be many instances of relationships between many instances of resources.

Note that this capability is not limited to manageable resources and can be exposed by any resource that wants to expose relationships that it knows about.

A relationship may become stale. The information about a relationship should be validated, either manually or automatically, before it can be relied upon. Exposing the information about a relationship should be considered a potential security risk if a participating resource should not be visible for security reasons.

4.1.2 Information Markup Declarations

4.1.2.1 Representation of Categories of Relationships

A relationship may be categorized as a certain type of relationship. A relationship type defines the semantics of the relationship. One relationship type may be a specialization or generalization of another type. This defines a taxonomy of relationship categories. MUWS defines a way to represent a type and its taxonomy lineage, but the actual definition of a relationship type is specific to a resource management model. Therefore, no relationship type is defined by MUWS.

In other words, MUWS specifies only the mechanism to convey a relationship type, or category, in XML as follows.

RelationshipTypeType type is declared as follows

```
<xs:complexType name=”RelationshipTypeType”>
  <xs:complexContent>
    <xs:extension base=”muws2:CategoryType”/>
  </xs:complexContent>
</xs:complexType>
```

The *RelationshipTypeType* type is used to declare an XML element containing instances of relationship type information.

The relationship type information MUST be declared as follows:

- An XML element declaring which QName identifies the semantics of a relationship type.
- The XML element MUST be declared with an XML Schema type that is a restriction of *RelationshipTypeType*.
- The contents of the XML element MUST be either
    - The only one XML element corresponding to the generalization of the currently declared relationship type.
    - The empty sequence, if the currently declared relationship type does not have a generalization, such as the top of a taxonomy.

For example, the “USB attached” relationship type may be generalized to the “Bus connected” type which, in turn, may be generalized to the “Generally linked” type. An instance of the “USB attached” relationship type information may be represented in the following XML fragment by using the rules described above:

```
<my:RelationshipTypeInstanceElement xsi:type=”RelationshipTypeType”>
  <usb:Attached>
    <bus:Connected>
      <generally:Linked/>
    <bus:Connected>
  </usb:Attached>
</my:RelationshipTypeInstanceElement>
```

4.1.2.2 Representation of an Instance of a Relationship

MUWS defines the following Global Element Declaration (GED) to represent an instance of a relationship.

```
<muws2:Relationship>
  <muws2:Name>xs:string</muws2:Name> ?
  <muws2:Type>muws2:RelationshipTypeType</muws2:Type>
  <muws2:Participant>
    <muws2:Self/> ?
    <muws1:ManageabilityEndpointReference/> *
    <wsa:EndpointReference/> *
    <muws1:ResourceId/> ?
    <muws2:Role>xs:anyURI</muws2:Role>
    {any} *
  </muws2:Participant>
  <muws2:Participant/>+
  <muws2:AccessEndpointReference>
    wsa:EndpointReferenceType
  </muws2:AccessEndpointReference>?
  {any} *
</muws2:Relationship>
```

muws2:Relationship/muws2:Name is a human readable name for a relationship. *Name* should not be used for machine reasoning about the semantics of a relationship. Type should be used instead. This element is OPTIONAL.

muws2:Relationship/muws2:Type is the relationship type this relationship belongs to. Examples of such types include linkage, containment, or dependency. MUWS does not define any specific relationship type. This is left to domain-specific models. MUWS only defines a way to convey the type as part of the representation of a relationship. In order to allow relationships to be defined as part of a taxonomy, the mechanism used by MUWS to represent relationship types leverages the *muws2:CategoryType* type defined in section 2.6. This element is REQUIRED.

muws2:Relationship/muws2:Participant contains information about a participant in the relationship. There MUST be at least two participants, but there MAY be more than two participants.

muws2:Relationship/muws2:Participant/muws2:Self: is an optional empty element that allows a participant to reference its self as the participant.

muws2:Relationship/muws2:Participant/muws1:ManageabilityEndpointReference is a reference to a WSDM manageability endpoint. This GED is defined in part 1. It MAY be included if a participant is a WSDM manageable resource and the provider wishes to expose this information. If more than one manageability endpoint is known, then more than one instance of this element MAY be present.

muws2:Relationship/muws2:Participant/wsa:EndpointReference is a WS-Addressing endpoint Reference.

muws2:Relationship/muws2:Participant/muws1:ResourceId is a WSDM manageable resource identifier which MAY be reported by the provider of relationship information. This GED is defined in part 1. This information may be used to locate manageability endpoints for a participant, or may be used for other purposes. For example, a resource identifier SHOULD be used to express that the provider of relationship information is also a participant in a relationship by returning its own resource identifier as one of the participants. Obviously, in order for this assertion to work, the provider of relationship information must be a WSDM manageable resource.

muws2:Relationship/muws2:Participant/muws2:Role is a URI which identifies the role a participant plays in a relationship. A participant role MUST be unique within a given instance of the relationship. The set of valid roles is defined by a relationship type. This attribute is REQUIRED.

muws2:Relationship/muws2:Participant/{any}* is an XML extensibility content which MAY contain elements that further or otherwise describe a participant.
muws2:Relationship/muws2:AccessEndpoint is a reference to a Web service endpoint which provides access to this relationship (if available). The endpoint MUST implement the relationship resource capability (see section 4.2).

The following is an example of a relationship information instance. The relationship is a WSDM manageable network host myhost.myorg.org containing an attached SCSI disk. The SCSI disk is exposed as a functional or operational endpoint of a Web service (e.g. to read/write from the disk). The “containment” relationship is represented by the following XML instance fragment:

```
<muws2:Relationship>
  <muws2:Name>SCSI disk attached to the host computer</muws2:Name>
  <muws2:Type>
    <scsi:Attached>
      <bus:Connected>
        <generally:Linked/>
      </bus:Connected>
    </scsi:Attached>
  </muws2:Type>
  <muws2:Participant>
    <muws1:ManageabilityEndpointReference>
```

```
      ...EPR1...
    </muws1:ManageabilityEndpointReference>
    <wsa:EndpointRefence>
      ...EPR2...
    </wsa:EndpointReference>
    <muws1:ResourceId>urn:uuid:123</muws1:ResourceId>
    <muws2:Role>urn:role:bus:host</muws2:Role>
    <netop-xs:HostName>myhost.myorg.org</netop-xs:NostName>
  </muws2:Participant>
  <muws2:Participant>
    <muws2:Self/>
    <muws2:Role>urn:role:bus:device</muws2:Role>
    <scsi-xs:Port>2</scsi-xs:Port>
    <scsi-xs:CH>0</scsi-xs:CH>
    <scsi-xs:BusID>5</scsi-xs:BusID>
    <scsi-xs:LUN>0</scsi-xs:LUN>
  </muws2:Participant>
</muws2:Relationship>
```

4.1.3 Properties

The Relationship capability defines the following property:

```
<muws2:Relationship/> *
```

muws2:Relationship is a representation of a relationship of which the provider of this capability is aware. See section 4.1.2.2 for the definition of the Relationship element. The provider of this capability is not necessarily a participant in any relationship represented by this property.

It is not recommended to request all values of the Relationship property with either *wsrf-rp:GetResourceProperty* or *wsrf-rp:GetMultipleResourceProperties* operations as there may be too many relationships. The use of the wsrf-rp:QueryResourceProperties operation is RECOMMENDED when retrieving the Relationships property. A provider of this manageability capability SHOULD, in general, support the *wsrf-rp:QueryResourceProperties* operation. However, if the provider of this capability knows of just a few relationships, it MAY choose not to support *wsrf-rp:QueryResourceProperties* operation.

For example, the following request may be sent to retrieve all “Bus connected” relationships which point to devices exposed as Web services.

```
<soap:Envelope ...>
  <soap:Header>
       ...
  </soap:Header>
  <soap:Body>
    <wsrf-rp:QueryResourceProperties>
      <wsrf-rp:QueryExpression
        Dialect=”http://www.w3.org/TR/1999/REC-xpath-19991116” >
boolean(/*/muws2:Relationship/muws2:Type/*/bus:Connected and
/*/muws2:Relationship/muws2:Participant[Role=”urn:role:bus:device”]/mows-
xs:EndpointReference)
      </wsrf-rp:QueryExpression>
    </wsrf-rp:QueryResourceProperties>
  </soap:Body>
</soap:Envelope>
```

4.1.4 Operations

This capability defines the following message exchanges.

4.1.4.1 QueryRelationshipsByType

This operation is OPTIONAL. It is a shortcut to query relationships of the same type. The request to perform this operation has a payload as follows:

```
<muws2:QueryRelationshipsByType>
       <muws2:RequestedType>xs:QName</muws2:RequestedType> +
</muws2:QueryRelationshipsByType>
```

muws2:QueryRelationshipsByType is a Global Element Declaration (GED) which identifies the operation requested.

muws2:QueryRelationshipsByType/muws2:RequestedType is a QName which identifies the requested type(s) of relationship(s). When processing this request, the manageability endpoint MUST return any available instance relationship that is of the requested type or of any type that is a specialization of the requested type. There can be more than one requested type, in which case any relationship instance corresponding to any requested type MUST be returned.

The response to the above request is either a fault (any fault) or the following message:

```
<muws2:QueryRelationshipsByTypeResponse>
       <muws2:Relationship/> *
</muws2:QueryRelationshipsByTypeResponse>
```

muws2:QueryRelationshipsByTypeResponse is a GED which identifies a response to the requested operation.

muws2:QueryRelationshipsByTypeResponse/muws2:Relationship is a relationship representation matching a requested type. There is one such element for each relationship instance corresponding to at least one requested type.

This operation has the following *Capability* metadata item:

```
<muws2:Capability>
  http://docs.oasis-open.org/wsdm/muws/capabilities/Relationships
</muws2:Capability>
```

4.1.5 Events

To support notifications on a change in a relationship, the following notification topics are defined in the relationships capability:

```
<wstop:Topic name=”RelationshipCreated”
messageTypes=”muws2:RelationshipCreatedNotification””>
  <wstop:MessagePattern
    Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">//muws1:ManagementEvent[count(muws2:RelationshipCreatedNotificati
on)=1]
  </wstop:MessagePattern>
</wstop:Topic>
<wstop:Topic name=”RelationshipDeleted”
messageTypes=”muws2:RelationshipDeletedNotification”>
  <wstop:MessagePattern
    Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">//muws1:ManagementEvent[count(muws2:RelationshipDeletedNotificati
on)=1]
  </wstop:MessagePattern>
</wstop:Topic>
```

muwse:RelationshipCreated indicates the addition of a new relationship. It is RECOMMENDED that a consumer subscribe to this notification with an appropriate selector against the content of notification messages in order to reduce the volume of received messages. Each notification message contains at least the following information:

```
<RelationshipCreatedNotification>
  <Relationship/>
</RelationshipCreatedNotification>
```

muwse:RelationshipDeleted indicates removal of an existing relationship. It is RECOMMENDED that a consumer subscribe to this notification with an appropriate selector against the content of notification messages in order to reduce the volume of received messages. Each notification message contains at least the following information:

```
<RelationshipDeletedNotification>
  <Relationship/>
</RelationshipDeletedNotification>

```

4.2 Relationship Resource Capability

The manageability capability URI for this capability is
http://docs.oasis-open.org/wsdm/muws/capabilities/RelationshipResource

4.2.1 Definition

A Web service endpoint, in addition to providing access to a relationship may also represent a relationship. Representing a relationship means that an endpoint is able to provide relationship information as described in section 4.1.2.2. In this case, a Web service endpoint MUST be a WS-Resource, as defined by the WSRF. One such WS-Resource provides information about one relationship instance. Representing a relationship as a WS-Resource is useful when a manageability model defines additional properties, operations or events for a relationship.

In order to represent a relationship as a WS-Resource, a set of properties is normatively required. The rest of the representation depends upon the relationship manageability model and discretion of the provider of a WS-Resource and relationship.

4.2.2 Properties

The Relationship Resource capability defines the following properties.

```
<muws2:Name>xs:string</muws2:Name> ?
```

muws2:Name is an element as defined by the Relationship/Name in section 4.1.2.2. It is OPTIONAL.

```
<muws2:Type>muws2:RelationshipTypeType</muws2:Type>
```

muws2:Type is an element as defined by the Relationship/Type in section 4.1.2.2. It is REQUIRED and can only appear once.

```
<muws2:Participant>
  <muws1:ManageabilityEndpointReference/> *
  <wsa:EndpointReference/> *
  <muws1:ResourceId/> ?
  <muws2:Role>xs:anyURI</muws2:Role>
  {any} *
</muws2:Participant>
```

muws2:Participant is an element as defined by the Relationship/Participant in section 4.1.2.2. This element MUST appear at least twice, and exactly once per participant in the relationship.

Even though the optional Self element is still in the Participant element schema, the Self element should not be specified in participants of relationships when they are resources, as shown in the example above.

4.2.3 Events

The *muwse:RelationshipResourceCapability* topic defined below is used for events related to the *Relationship Resource* capability.

```
<wstop:Topic name="RelationshipResourceCapability"
             messageTypes="muws1:ManagementEvent">
</wstop:Topic>
```

4.3 Advertisement

The manageability capability URI for the Advertisement capability is http://docs.oasis-open.org/wsdm/muws/capabilities/Advertisement

4.3.1 Definition

The *Advertisement* capability is exposed by a Web service that is able to provide a notification on the creation or the destruction of a manageable resource. Since a consumer cannot register for a notification on a resource before the resource is created, a creation event is reported for some other resource by the implementer of a “lifetime notification” capability.

Note that this capability may be implemented by a manageable resource or by some other service (see section 4 on the distinction between “manageability capability” and “management-related capability”. A service might offer a capability to notify on the creation or the destruction of a resource even though the service itself is not manageable. For example, if a system includes a registry, to which a resource is added as soon as it is created, and from which it is removed when it is destroyed, then this registry could expose the *Advertisement* capability and use it to share information about resource creation and destruction events with manageability consumers. Likewise, a resource factory might emit creation events for a resource it creates, yet the factory itself might not be manageable. Another example is a container, a J2EE server or a business process execution engine for example, that can send a notification when a contained resource is created.

This capability defines four topics used for notification but does not define any property or operation.

In addition to advertisement by sending notifications, as defined in this capability, another approach for advertisement is to register a manageable resource in a registry. A resource advertised in this way can be discovered using the mechanisms introduced in section 5.2.

4.3.2 Events

The Advertisement capability defines four notification topics:

```
<wstop:Topic name=”ManageabilityEndpointCreation”
messageTypes=”muws2:CreationNotification”>
  <wstop:MessagePattern
    Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">//muws1:ManagementEvent[count(muws2:CreationNotification)=1]
  </wstop:MessagePattern>
  <wstop:Topic name=”ManageableResourceCreation”
messageTypes=”muws2:CreationNotification”>
    <wstop:MessagePattern
      Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">//muws1:ManagementEvent[count(muws2:CreationNotification)=1]
    </wstop:MessagePattern>
  </wstop:Topic>
</wstop:Topic>
  <wstop:Topic name=”ManageabilityEndpointDestruction”
messageTypes=”muws2:DestructionNotification”>
  <wstop:MessagePattern
    Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">//muws1:ManagementEvent[count(muws2:DestructionNotification)=1]
  </wstop:MessagePattern>
  <wstop:Topic name=”ManageableResourceDestruction”
messageTypes=”muws2:DestructionNotification”/>
    <wstop:MessagePattern
      Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">//muws1:ManagementEvent[count(muws2:DestructionNotification)=1]
    </wstop:MessagePattern>
  </wstop:Topic>
</wstop:Topic>
```

The “**muwse:ManageabilityEndpointCreation**” topic corresponds to notification on the creation of a new manageability endpoint for a new or existing resource. A manageability endpoint may be created in conjunction with, or independent of, the creation of the manageable resource. A new manageability endpoint could be the first one for a resource or be an addition to others. An associated *muws2:CreationNotification* message contains the EPR of a newly created manageability endpoint.

The “**muwse:ManageableResourceCreation**” topic is a specialization of the “Manageability EndpointCreation” topic. This topic corresponds to the case where a resource itself is newly created. Note that if a resource is created that is not manageable (i.e. which does not have a manageability endpoint) no notification on this topic will be sent. If a resource and a manageability endpoint for the resource are created then a notification will be sent to a subscriber on this topic.

The “**muwse:ManageabilityEndpointDestruction**” topic corresponds to notification on the destruction of a manageability endpoint. It does not imply that the associated resource was destroyed. An associated *muws2:DestructionNotification* message contains the *muws2:ResourceId* that a newly destroyed manageability endpoint provided for the resource before its destruction.

The “**muwse:ManageableResourceDestruction**” topic is a specialization of the “ManageabilityEndpointDestruction” topic. This topic corresponds to the case where a resource itself is destroyed at the same time as the manageability endpoint. Note that if a resource is destroyed that is not manageable (i.e. which does not have a manageability endpoint) no notification on this topic will be sent. An associated *muws2:DestructionNotification* message contains the *muws2:ResourceId* that a newly destroyed manageability endpoint provided for the resource before its destruction.

The content element for these topics are described as follows:

```
<muws2:CreationNotification">
  <muws1:ManageabilityEndpointReference"/> *
</muws2:CreationNotification">
```

muws2:CreationNotification/muws1:ManageabilityEndpointReference is a reference to the manageability endpoint of a newly created resource. There can be more than one such reference if there is more than one known manageability endpoint.

```
<muws2:DestructionNotification">
  <muws1:ResourceId"/> ?
</muws2:DestructionNotification">
```

muws2:DestructionNotification/muws1:ResourceId is the *ResourceId* of a newly destroyed resource.

5 Discovery

Many forms of discovery are supported by Web services. This specification does not prescribe a normative method for discovering manageability services. It is expected that discovery methods commonly used for Web services will be used as discovery methods for manageability services. The goal of discovery is to obtain the EPR of a manageability endpoint. The Advertisement capability (section 4.3), when supported, provides one way to facilitate discovery via events. This section also describes two other ways to discover manageable resources. These are just some of the discovery methods that can be used.

The only normative requirement relative to discovering manageability services is that a manageability service MUST provide the Identity capability as defined by MUWS. As a result of this requirement, a consumer can inspect the WSDL description for a Web service or attempt to use the Identity capability of a Web service to determine if a discovered service acts as a manageability service. If a discovered service provides at least the Identity capability as defined by MUWS, then it is a manageability service.

5.1 Discovery using Relationships

There are at least two scenarios in which a relationship can be used to discover a manageable resource.

The first scenario is when a manageable resource points to some other manageable resource through a relationship. A manageable resource that supports the Relationship capability enables discovery of an EPR for some other resource that participates in a relationship with the manageable resource. This is done by using the “Relationship” property defined in section 4.1.3 or invoking the operations defined in section 4.1.4. Any EPRs contained in such a response message may be used by the manageability consumer to disambiguate a manageable resource in an exchange of messages with a manageability endpoint.

The second scenario is when a consumer has access to a WS-Resource representing a relationship and the relationship has a manageable resource as a member. A consumer can then use the properties of the Relationship Resource capability to retrieve any EPRs of a manageable resource participating in the relationship.

5.2 Discovery using Registries

In addition to emitting a notification on the creation and the destruction of a resource as defined by the Advertisement capability in section 4.3, a resource can be advertised to a registry by invoking an insertion interface of the registry. A consumer can then discover a manageable resource by invoking a query interface of the registry.

The WSRF WS-Service Group specification **[WS-SG]** defines a type of registry, along with the message exchanges used to interact with a registry of this type. It is RECOMMENDED that a registry used to discover a manageable resource conforms to the WS-Service Group specification and that the registry conform to the following additional constraints:

The service group SHOULD include as properties the following two elements:

```
<wssg:MembershipContentRule
  MemberInterface=”muws1:Identity”
  ContentElements=”muws1:ResourceId”>
<wssg:MembershipContentRule
  MemberInterface=”muws1:ManageabilityCharacteristics”
  ContentElements=”muws1:ManageabilityCapability”>
```

The service group MAY also have any other “MembershipContentRule”, including a rule with an empty value for both MemberInterface and ContentElements. In effect, this lifts any constraint on a member of the service group. The two membership content rules defined above are useful even in a service group with no effective constraint because they allow querying the service group on the “ResourceId” and “ManageabilityCapability” properties.

When adding a manageability endpoint for a resource to the membership of a service group using the “Add” operation, the requestor SHOULD include the *muws1:ResourceId* element of a manageable resource in a *wssg:Add/wssg:Content* element of a request, even if the service group supports additional membership content rules that would have permitted registration of a manageability endpoint in the service group without providing this content element. Similarly, if the manageable resource supports the Manageability Characteristics capability, then the consumer SHOULD include all the *muws1:ManageabilityCapability* elements of a manageable resource in a *wssg:Add/wssg:Content* element of a request, even if the service group supports additional membership content rules that would have permitted registration of the manageability endpoint in the service group without providing this content element.

Like any manageability endpoint, a manageability endpoint listed in a resource registry MUST implement the Identity capability defined in **[MUWS Part 1]**. In addition, in order to facilitate discovery, the manageability endpoint SHOULD implement the Manageability Characteristics capability as defined in **[MUWS Part 1]**.

6 References

6.1 Normative

[MUWS Part 1] Vaughn Bullard, *Web Services Distributed Management: Management using Web Services (MUWS 1.1) Part 1*, OASIS Committee Draft, March 2006, http://docs.oasis-open.org/wsdm/wsdm-muws1-1.1-spec-os-01.pdf

[XML1.0 3rd Edition]
Tim Bray, et al., *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C Recommendation, February 2004, http://www.w3.org/TR/REC-xml

[XML Schema Part 1]
Henry S. Thompson, et al. *XML Schema Part 1: Structures*, W3C Recommendation, May 2001, http://www.w3.org/TR/xmlschema-1/

[XML Schema Part 2]
Paul V. Biron, et al. *XML Schema Part 2: Datatypes*, W3C Recommendation, May 2001, http://www.w3.org/TR/xmlschema-2/

[WSDL] Erik Christensen, et al., *Web services Description Language (WSDL) 1.1*, W3C Note, March 2001, http://www.w3.org/TR/wsdl

[WS-Resource] Steve Graham, et al. *Web Service Resource 1.2 (WS-Resource)*, OASIS Standard, April 2006, http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf

[WS-Addressing] Don Box, et al., *Web services Addressing (WS-Addressing)*, W3C Member Submission, August 2004, http://www.w3.org/TR/ws-addr-core

[WS-RP] Steve Graham, et al., *Web Services Resource Properties 1.2 (WS-ResourceProperties)*, OASIS Standard, April 2006, http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-os.pdf

[XPath 1.0] James Clark, et al., *XML Path Language (XPath) Version 1.0*, W3C Recommendation, November 1999, http://www.w3.org/TR/1999/REC-xpath-19991116

[WSN] Steve Graham, et al., *Web Services Base Notification 1.2 (WS-BaseNotification)*, OASIS Public Review Draft, June 2006, http://www.oasis-open.org/apps/org/workgroup/wsn/wsn-ws_base_notification-1.3-spec-pr-03.pdf

[WST] William Vambenepe, *Web Services Topics 1.2 (WS-Topics)*, OASIS Public Review Draft, June 2006, http://www.oasis-open.org/apps/org/workgroup/wsn/wsn-ws_topics-1.3-spec-pr-02.pdf

[RFC3066] IETF (Internet Engineering Task Force). RFC 3066*: Tags for the Identification of Languages*, ed. H. Alvestrand. 2001, http://www.ietf.org/rfc/rfc3066.txt

[WS-RL] Latha Srinivasan, et al., *Web Services Resource Lifetime 1.2 (WS-ResourceLifetime)*, OASIS Standard, April 2006, http://docs.oasis-open.org/wsrf/wsrf-ws_resource_lifetime-1.2-spec-os.pdf

[WS-SG] Tom Maguire, et al., *Web Services Service Group 1.2 (WS-ServiceGroup)*, OASIS Standard, April 2006, http://docs.oasis-open.org/wsrf/wsrf-ws_service_group-1.2-spec-os.pdf

6.2 Non-normative

[MOWS] Kirk Wilson, *Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.1*, OASIS Committee Draft, February 2006, http://docs.oasis-open.org/wsdm/wsdm-mows-1.1-spec-os-01.pdf

[SOAP] Don Box, et al., *Simple Object Access Protocol (SOAP) 1.1*, W3C Note, May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[WS-RF] WSRF OASIS technical committee, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf

[WSRMD] Steve Graham, et al., *Web Services Resource Metadata 1.0 (WS-ResourceMetadataDescriptor)*, OASIS Working Draft, January 2006, http://www.oasis-open.org/ wsrf/wsrf-ws_resource_metadata_descriptor-1.0-spec-06.pdf

Appendix A. Acknowledgements

WSDM Management Using Web Services Part 2 Version 1.0 Acknowledgements

The following individuals were members of the committee when the WSDM MUWS Version 1.0 was approved by the technical committee

Guru Bhat, Jeff Bohren, Winston Bumpus, Nick Butler, Brian Carroll, Fred Carter, Michael Clements, David Cox, John DeCarlo, Andreas Dharmawan, Mark Ellison, John Fuller, Paul Lipton, Heather Kreger, Hal Lockhart, Frederico Maciel, Tom Maguire, Bryan Murray, Richard Nikula, Mark Peel, Richard Pelavin, Homayoun Pourheidari, Warren Roberts, Karl Schopmeyer, Igor Sedukhin, David Snelling, Thomas Studwell, William Vambenepe, Andrea Westerinen, Jim Willits, Zhili Zhang.

In addition, the following non-member employees of member companies made contribution to the specification: Maryann Hondo, Ian Springer, John Gerken, David Melgar, Mitsunori Satomi.

WSDM Management Using Web Services Part 2 Version 1.1 Acknowledgements

The following people made contributions to the WSDM MUWS Version 1.1 specification: Vaughn Bullard, Fred Carter, David Cox, Zulah Eckert, Mark Ellison, Heather Kreger, Frederico Maciel, Bryan Murray, Richard Nikula, Mitsunori Satomi, Thomas Studwell, Kirk Wilson, Zhili Zhang with special thanks to Vaughn Bullard and Mark Ellison as editors.

The following individuals were members of the committee while the WSDM MUWS Version 1.1 specification was developed and approved by the technical committee: Guru Bhat, Jeff Bohren, Vaughn Bullard, Winston Bumpus, Fred Carter, Michael Clements, David Cox, Zulah Eckert, Mark Ellison, John Fuller, Tony Gullato, Heather Kreger, Richard Landau, Frederico Maciel, Tom Maguire, David Melgar, Bryan Murray, Richard Nikula, Mark Peel, Mitsunori Satomi, Thomas Studwell, William Vambenepe, Kirk Wilson, Zhili Zhang.

Appendix B. 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 implementers 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.

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 does 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.

Appendix C. Schemas

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema targetNamespace="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
       xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
       <xs:import namespace="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
schemaLocation="http://docs.oasis-open.org/wsdm/muws1-2.xsd" />
       <xs:import namespace="http://www.w3.org/2005/08/addressing"
schemaLocation="http://www.w3.org/2005/08/addressing/ws-addr.xsd" />
       <xs:complexType name="LangString">
              <xs:simpleContent>
                     <xs:extension base="xs:string">
                            <xs:attribute ref="xml:lang" use="required" />
                            <xs:anyAttribute namespace="##other" />
                     </xs:extension>
              </xs:simpleContent>
       </xs:complexType>
       <!-- Begin properties for the Description capability -->
       <xs:element name="Caption" type="muws2:LangString" />
       <xs:element name="Description" type="muws2:LangString" />
       <xs:element name="Version" type="xs:string" />
       <!-- End   properties for the Description capability -->
       <xs:complexType name="CategoryType">
              <xs:sequence>
                     <xs:any minOccurs="0" namespace="##any"
processContents="lax" />
              </xs:sequence>
       </xs:complexType>
       <xs:complexType name="StateType">
              <xs:complexContent>
                     <xs:extension base="muws2:CategoryType" />
              </xs:complexContent>
       </xs:complexType>
       <xs:element name="State" type="muws2:StateType" />
       <xs:element name="EnteredState" type="muws2:StateType" />
       <xs:element name="PreviousState" type="muws2:StateType" />
       <xs:complexType name="StateTransitionType">
              <xs:sequence>
                     <xs:element ref="muws2:EnteredState" />
                     <xs:element ref="muws2:PreviousState" minOccurs="0" />
                     <xs:any minOccurs="0" maxOccurs="unbounded"
namespace="##other" processContents="lax" />
              </xs:sequence>
              <xs:attribute name="TransitionIdentifier" type="xs:anyURI"
use="optional" />
              <xs:attribute name="Time" type="xs:dateTime" use="required" />
              <xs:anyAttribute namespace="##other" />
       </xs:complexType>
       <xs:element name="StateTransition" type="muws2:StateTransitionType" />
       <!-- Begin properties for the OperationalStatus capability -->
       <xs:element name="OperationalStatus">
              <xs:simpleType>
                     <xs:restriction base="xs:string">
                            <xs:enumeration value="Available" />
                            <xs:enumeration value="PartiallyAvailable" />
                            <xs:enumeration value="Unavailable" />
                            <xs:enumeration value="Unknown" />
```

```
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <!-- End   properties for the OperationalStatus capability -->
    <xs:attributeGroup name="MetricAttributes">
        <xs:attribute name="ResetAt" type="xs:dateTime" />
        <xs:attribute name="LastUpdated" type="xs:dateTime" />
        <xs:attribute name="Duration" type="xs:duration" />
    </xs:attributeGroup>
    <!-- Begin properties for the Metrics capability -->
    <xs:element name="CurrentTime" type="xs:dateTime" />
    <!-- End   properties for the Metrics capability -->
    <xs:complexType name="RelationshipTypeType">
        <xs:complexContent>
            <xs:extension base="muws2:CategoryType" />
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="Self">
        <xs:complexType />
    </xs:element>
    <xs:complexType name="RelationshipParticipantType">
        <xs:sequence>
            <xs:element ref="muws2:Self" minOccurs="0" />
            <xs:element ref="muws1:ManageabilityEndpointReference"
minOccurs="0" maxOccurs="unbounded" />
            <xs:element ref="wsa:EndpointReference" minOccurs="0"
maxOccurs="unbounded" />
            <xs:element ref="muws1:ResourceId" minOccurs="0" />
            <xs:element name="Role" type="xs:anyURI" />
            <xs:any minOccurs="0" maxOccurs="unbounded"
namespace="##other" processContents="lax" />
        </xs:sequence>
        <xs:anyAttribute namespace="##other" />
    </xs:complexType>
    <!-- Begin properties for the RelationshipResource capability -->
    <xs:element name="Name" type="xs:string" />
    <xs:element name="Type" type="muws2:RelationshipTypeType" />
    <xs:element name="Participant" type="muws2:RelationshipParticipantType"
/>
    <!-- End   properties for the RelationshipResource capability -->
    <xs:complexType name="RelationshipType">
        <xs:sequence>
            <xs:element ref="muws2:Name" minOccurs="0" />
            <xs:element ref="muws2:Type" />
            <xs:element ref="muws2:Participant" minOccurs="2"
maxOccurs="unbounded" />
            <xs:element name="AccessEndpointReference"
type="wsa:EndpointReferenceType" minOccurs="0" />
            <xs:any minOccurs="0" maxOccurs="unbounded"
namespace="##other" processContents="lax" />
        </xs:sequence>
        <xs:anyAttribute namespace="##other" />
    </xs:complexType>
    <!-- Begin properties for the Relationship capability -->
    <xs:element name="Relationship" type="muws2:RelationshipType" />
    <!-- End   properties for the Relationship capability -->
    <xs:element name="RelationshipCreatedNotification">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="muws2:Relationship" />
                <xs:any minOccurs="0" maxOccurs="unbounded"
namespace="##other" processContents="lax" />
            </xs:sequence>
```

```
                  <xs:anyAttribute namespace="##other" />
            </xs:complexType>
      </xs:element>
      <xs:element name="RelationshipDeletedNotification">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element ref="muws2:Relationship" />
                        <xs:any minOccurs="0" maxOccurs="unbounded"
namespace="##other" processContents="lax" />
                  </xs:sequence>
                  <xs:anyAttribute namespace="##other" />
            </xs:complexType>
      </xs:element>
      <xs:element name="QueryRelationshipsByType">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element name="RequestedType" type="xs:QName" />
                  </xs:sequence>
            </xs:complexType>
      </xs:element>
      <xs:element name="QueryRelationshipsByTypeResponse">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element ref="muws2:Relationship" minOccurs="0"
maxOccurs="unbounded" />
                  </xs:sequence>
            </xs:complexType>
      </xs:element>
      <xs:element name="CreationNotification">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element
ref="muws1:ManageabilityEndpointReference" minOccurs="0" maxOccurs="unbounded"
/>
                  </xs:sequence>
                  <xs:anyAttribute namespace="##other" />
            </xs:complexType>
      </xs:element>
      <xs:element name="DestructionNotification">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element ref="muws1:ResourceId" minOccurs="0" />
                  </xs:sequence>
                  <xs:anyAttribute namespace="##other" />
            </xs:complexType>
      </xs:element>
      <xs:complexType name="SituationCategoryType">
            <xs:complexContent>
                  <xs:extension base="muws2:CategoryType" />
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="SubstitutableMsgType">
            <xs:sequence>
                  <xs:element name="Value" type="xs:anySimpleType"
minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="MsgId" type="xs:string" use="required" />
            <xs:attribute name="MsgIdType" type="xs:anyURI" use="required" />
      </xs:complexType>
      <xs:complexType name="SituationType">
            <xs:sequence>
                  <xs:element name="SituationCategory"
type="muws2:SituationCategoryType" />
```

```
            <xs:element name="SuccessDisposition" minOccurs="0">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="Successful" />
                        <xs:enumeration value="Unsuccessful" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="SituationTime" type="xs:dateTime" />
            <xs:element name="Priority" type="xs:short" minOccurs="0"
/>
            <xs:element name="Severity" type="xs:short" minOccurs="0"
/>
            <xs:element name="Message" type="muws2:LangString"
minOccurs="0" />
            <xs:element name="SubstitutableMsg"
type="muws2:SubstitutableMsgType" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
    <xs:element name="Situation" type="muws2:SituationType" />
    <xs:complexType name="EventCorrelationPropertiesType">
        <xs:sequence>
            <xs:element name="repeatCount" minOccurs="0" maxOccurs="1">
                <xs:simpleType>
                    <xs:restriction base="xs:short">
                        <xs:minInclusive value="0" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="elapsedTime" minOccurs="0" maxOccurs="1">
                <xs:simpleType>
                    <xs:restriction base="xs:long">
                        <xs:minInclusive value="0" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="sequenceNumber" type="xs:unsignedLong"/>
    </xs:complexType>
    <xs:element name="EventCorrelationProperties"
type="muws2:EventCorrelationPropertiesType" />
    <xs:complexType name="MsgCatalogInformationType">
        <xs:sequence>
            <xs:element name="msgCatalog" type="xs:anyURI"
minOccurs="1" />
            <xs:element name="msgCatalogType" type="xs:anyURI"
minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
    <xs:element name="MsgCatalogInformation"
type="muws2:MsgCatalogInformationType" />
    <!-- #####   Metadata description elements   ##### -->
    <xs:element name="Capability" type="xs:anyURI" />
    <xs:complexType name="DialectableExpressionType" mixed="true">
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax"
minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="Dialect" type="xs:anyURI" use="required" />
        <xs:anyAttribute namespace="##other" />
    </xs:complexType>
    <xs:element name="ValidWhile" type="muws2:DialectableExpressionType" />
    <xs:element name="Units" type="xs:string" />
```

```
        <xs:element name="ChangeType">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                                <xs:enumeration value="Counter" />
                                <xs:enumeration value="Gauge" />
                                <xs:enumeration value="Unknown" />
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="TimeScope">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                                <xs:enumeration value="Interval" />
                                <xs:enumeration value="PointInTime" />
                                <xs:enumeration value="SinceReset" />
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="GatheringTime">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                                <xs:enumeration value="OnChange" />
                                <xs:enumeration value="Periodic" />
                                <xs:enumeration value="OnDemand" />
                                <xs:enumeration value="Unknown" />
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="CalculationInterval" type="xs:duration" />
        <xs:element name="MetricGroup" type="xs:anyURI" />
        <xs:element name="PostCondition" type="muws2:DialectableExpressionType"
/>
        <!-- =========   StartSituation   ============  -->
        <xs:element name="StartSituation">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType" />
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <xs:element name="StartInitiated">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType">
                                        <xs:sequence>
                                                <xs:element ref="muws2:StartSituation"
/>
                                        </xs:sequence>
                                </xs:restriction>
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <xs:element name="RestartInitiated">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType">
                                        <xs:sequence>
                                                <xs:element ref="muws2:StartSituation"
/>
                                        </xs:sequence>
                                </xs:restriction>
                        </xs:complexContent>
                </xs:complexType>
```

```
       </xs:element>
       <xs:element name="StartCompleted">
              <xs:complexType>
                     <xs:complexContent>
                            <xs:restriction base="muws2:SituationCategoryType">
                                   <xs:sequence>
                                          <xs:element ref="muws2:StartSituation"
/>
                                   </xs:sequence>
                            </xs:restriction>
                     </xs:complexContent>
              </xs:complexType>
       </xs:element>
       <!-- =========   StopSituation   ============  -->
       <xs:element name="StopSituation">
              <xs:complexType>
                     <xs:complexContent>
                            <xs:restriction base="muws2:SituationCategoryType" />
                     </xs:complexContent>
              </xs:complexType>
       </xs:element>
       <xs:element name="StopInitiated">
              <xs:complexType>
                     <xs:complexContent>
                            <xs:restriction base="muws2:SituationCategoryType">
                                   <xs:sequence>
                                          <xs:element ref="muws2:StopSituation"
/>
                                   </xs:sequence>
                            </xs:restriction>
                     </xs:complexContent>
              </xs:complexType>
       </xs:element>
       <xs:element name="AbortInitiated">
              <xs:complexType>
                     <xs:complexContent>
                            <xs:restriction base="muws2:SituationCategoryType">
                                   <xs:sequence>
                                          <xs:element ref="muws2:StopSituation"
/>
                                   </xs:sequence>
                            </xs:restriction>
                     </xs:complexContent>
              </xs:complexType>
       </xs:element>
       <xs:element name="PauseInitiated">
              <xs:complexType>
                     <xs:complexContent>
                            <xs:restriction base="muws2:SituationCategoryType">
                                   <xs:sequence>
                                          <xs:element ref="muws2:StopSituation"
/>
                                   </xs:sequence>
                            </xs:restriction>
                     </xs:complexContent>
              </xs:complexType>
       </xs:element>
       <xs:element name="StopCompleted">
              <xs:complexType>
                     <xs:complexContent>
                            <xs:restriction base="muws2:SituationCategoryType">
                                   <xs:sequence>
```

```
                        <xs:element ref="muws2:StopSituation"
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <!-- =========   RequestSituation   ============  -->
    <xs:element name="RequestSituation">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType" />
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="RequestInitiated">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element
ref="muws2:RequestSituation" />
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="RequestCompleted">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element
ref="muws2:RequestSituation" />
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <!-- =========   DestroySituation   ============  -->
    <xs:element name="DestroySituation">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType" />
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="DestroyInitiated">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element
ref="muws2:DestroySituation" />
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="DestroyCompleted">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
```

```
                                        <xs:sequence>
                                                <xs:element 
ref="muws2:DestroySituation" />
                                        </xs:sequence>
                                </xs:restriction>
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <!-- =========   CreateSituation   ============  -->
        <xs:element name="CreateSituation">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType" />
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <xs:element name="CreateInitiated">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType">
                                        <xs:sequence>
                                                <xs:element ref="muws2:CreateSituation" 
/>
                                        </xs:sequence>
                                </xs:restriction>
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <xs:element name="CreateCompleted">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType">
                                        <xs:sequence>
                                                <xs:element ref="muws2:CreateSituation" 
/>
                                        </xs:sequence>
                                </xs:restriction>
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <!-- =========   ConnectSituation   ============  -->
        <xs:element name="ConnectSituation">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType" />
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <xs:element name="ConnectInitiated">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType">
                                        <xs:sequence>
                                                <xs:element 
ref="muws2:ConnectSituation" />
                                        </xs:sequence>
                                </xs:restriction>
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <xs:element name="ReconnectInitiated">
                <xs:complexType>
                        <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element 
ref="muws2:ConnectSituation" />
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="ConnectCompleted">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element 
ref="muws2:ConnectSituation" />
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <!-- =========   ReportSituation   ============  -->
    <xs:element name="ReportSituation">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType" />
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="PerformanceReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation" 
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="SecurityReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation" 
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="HeartbeatReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation" 
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="StatusReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation"
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="TraceReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation"
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="DebugReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation"
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="LogReport">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType">
                    <xs:sequence>
                        <xs:element ref="muws2:ReportSituation"
/>
                    </xs:sequence>
                </xs:restriction>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <!-- =========   AvailabilitySituation   ============  -->
    <xs:element name="AvailabilitySituation">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType" />
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <!-- =========   CapabilitySituation   ============  -->
    <xs:element name="CapabilitySituation">
        <xs:complexType>
            <xs:complexContent>
                <xs:restriction base="muws2:SituationCategoryType" />
```

```
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <!-- =========   ConfigureSituation   ============  -->
        <xs:element name="ConfigureSituation">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType" />
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <!-- =========   OtherSituation   ============  -->
        <xs:element name="OtherSituation">
                <xs:complexType>
                        <xs:complexContent>
                                <xs:restriction base="muws2:SituationCategoryType" />
                        </xs:complexContent>
                </xs:complexType>
        </xs:element>
        <!--
                                       SCHEMA COPY Material
Copy and paste element references below into the schema of a resource
properties document.
These references insure that the correct minOccurs/maxOccurs attributes are
specified in a resource property document schema.

NOTE: You must import the MUWS Part 2 schema namespace (MUWS2).

        **    Description Properties    **
            <xs:element ref="muws2:Caption"
                  minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="muws2:Description"
                  minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="muws2:Version"
                  minOccurs="0"/>

        **    Operational Status        **
            <xs:element ref="muws2:OperationalStatus"/>

        **    Metrics                   **
            <xs:element ref="muws2:CurrentTime"/>

        **    Relationship              **
            <xs:element ref="muws2:Relationship"
                  minOccurs="0" maxOccurs="unbounded"/>

        **    Relationship Resource     **
            <xs:element ref="muws2:Name" minOccurs="0"/>
            <xs:element ref="muws2:Type"/>
            <xs:element ref="muws2:Participant"
                  minOccurs="2" maxOccurs="unbounded"/>

-->
</xs:schema>
```

Appendix D. WSDL elements

```
<?xml version="1.0" encoding="utf-8"?>
<definitions
    targetNamespace="http://docs.oasis-open.org/wsdm/muws2-2.wsdl"
    xmlns:muws-p2-wsdl="http://docs.oasis-open.org/wsdm/muws1-2.wsdl"
    xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
    xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
    xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-1.xsd"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="http://schemas.xmlsoap.org/wsdl/">

  <types>
    <xs:schema elementFormDefault="qualified"
        targetNamespace="http://docs.oasis-open.org/wsdm/muws2-2.wsdl">

      <xs:import namespace="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
          schemaLocation="http://docs.oasis-open.org/wsdm/muws2-2.xsd"/>

      <xs:import namespace="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
          schemaLocation="http://docs.oasis-open.org/wsdm/muws1-2.xsd"/>

    </xs:schema>
  </types>


  <message name="QueryRelationshipsByTypeRequest">
    <part name="body" element="muws2:QueryRelationshipsByType"/>
  </message>

  <message name="QueryRelationshipsByTypeResponse">
    <part name="body" element="muws2:QueryRelationshipsByTypeResponse"/>
  </message>

</definitions>

<!--
                                       WSDL COPY Material
Copy and paste the operation specification below into a portType definition of
the WSDL documents of a web service.

NOTE: You must import the MUWS WSDL (wsdmmuws2-2).

         <operation name="QueryRelationshipsByType">
           <input name="QueryRelationshipsByTypeRequest"
                  message="muws-p2-wsdl:QueryRelationshipsByTypeRequest"/>
           <output name="QueryRelationshipsByTypeResponse"
                   message="muws-p2-wsdl:QueryRelationshipsByTypeResponse"/>
         </operation>

-->
```

Appendix E. Topics

```
<wstop:TopicSpace name="MuwsNotificationTopics"
    targetNamespace="http://docs.oasis-open.org/wsdm/muwse2-2.xml"
    xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
    xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
    xmlns:wstop="http://docs.oasis-open.org/wsn/t-1.xsd"
    xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-1.xsd">

  <wstop:Topic name="IdentityCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="ManageabilityCharacteristicsCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="CorrelatablePropertiesCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="DescriptionCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="StateCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="OperationalStatusCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="MetricsCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="ConfigurationCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="RelationshipsCapability"
               messageTypes="muws1:ManagementEvent">

    <wstop:Topic name="RelationshipCreated"
                 messageTypes="muws1:ManagementEvent">
      <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">
        //muws1:ManagementEvent[count(muws2:RelationshipCreatedNotification)=1]
      </wstop:MessagePattern>
    </wstop:Topic>

    <wstop:Topic name="RelationshipDeleted"
                 messageTypes="muws1:ManagementEvent">
      <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">
        //muws1:ManagementEvent[count(muws2:RelationshipDeletedNotification)=1]
      </wstop:MessagePattern>
    </wstop:Topic>

  </wstop:Topic>

  <wstop:Topic name="RelationshipResourceCapability"
               messageTypes="muws1:ManagementEvent">
  </wstop:Topic>

  <wstop:Topic name="ManageabilityEndpointCreation"
               messageTypes="muws1:ManagementEvent">
    <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">
      //muws1:ManagementEvent[count(muws2:CreationNotification)=1]
    </wstop:MessagePattern>

    <wstop:Topic name="ManageableResourceCreation"
                 messageTypes="muws1:ManagementEvent">
      <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">
        //muws1:ManagementEvent[count(muws2:CreationNotification)=1]
      </wstop:MessagePattern>
    </wstop:Topic>

  </wstop:Topic>

  <wstop:Topic name="ManageabilityEndpointDestruction"
               messageTypes="muws1:ManagementEvent">
    <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">
      //muws1:ManagementEvent[count(muws2:DestructionNotification)=1]
    </wstop:MessagePattern>

    <wstop:Topic name="ManageableResourceDestruction"
                 messageTypes="muws1:ManagementEvent">
      <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-
19991116">
        //muws1:ManagementEvent[count(muws2:DestructionNotification)=1]
      </wstop:MessagePattern>
    </wstop:Topic>

  </wstop:Topic>

</wstop:TopicSpace>
```

Appendix F. Description of situation types

This appendix defines in more details the situation types introduced in section 2.5.1.

AvailabilitySituation

This category deals with the situations reported from the component, regarding its operational state and availability. This situation provides a context for operations that can be performed by the component to establish if a product is installed, operational and ready to process functional requests, or operational and ready or not ready to process management requests. Existing message include words like “now ready to take requests”, “online”, and “offline”, for example::

- “SOAP connector available at port 8888”

CapabilitySituation

This category is specified when a change in capability of a resource occurs. For example, a printer has an envelope tray attached to it so that the printer is now has additional paper choices. The same category would be used if the envelope tray is removed from the printer.

ConfigurationSituation

This category deals with the components identifying configuration changes. Any changes that a component makes to its configuration should be logged using this category. Existing message include words like “port number is”, “address is”, and “process id”, for example:

- “File transfer configured with host=’9.27.11.13’, port=’9090’, securityEnabled=’false’”

StopSituation

This category deals with the shutdown process for a component. Messages that indicate that a component has begun to stop, that it has stopped, or that the stopping process has failed all fall into this category. Existing messages include words like “stop”, “stopping”, “stopped”, “completed”, and “exiting”, for example:

- “Application stopped: myApp.exe”
- “An error occurred while stopping myApp.exe”
- “Stopping the JMS provider”

StartSituation

This category deals with the startup process for a component. Messages that indicate that a component has begun the startup process, that it has finished the startup process, or that it has aborted the startup process all fall into this category. Existing messages include words like “starting”, “started”, “initializing”, and “initialized”, for example:

- “XYZ protocol support was successfully started”
- “XYZ protocol support failed to start”
- “Starting EJB: myEjb.jar”

RequestSituation

This category is used in situations that a component uses to identify the completion status of a request. Typically, these requests are complex management tasks or transactions that a component undertakes on behalf of a requestor and not the mainline simple requests or transactions. Existing messages are of the form “*request* started” or “*request* completed” as in phrases like “configuration synchronization started”, and “backup procedure complete”, for example:

- “Configuration synchronization completed”

Note that events generated from requests that start up or stop a resource would be categorized as **StartSituation** or **StopSituation** respectively because they are higher precedent than **RequestSituation**.

DestroySituation

This category deals with the situations occurring when an entity or component was removed or destroyed. Messages telling that a document was destroyed or a file was deleted all fall into this category. Existing messages include phrases like “was destroyed”, “about to remove”, and “no longer exists”, for example:

- “The connection pool was destroyed for data source foo”

CreateSituation

This category deals with the situations occurring when a component creates an entity. Messages telling that a document was created, or a file was created, or an Enterprise JavaBean (EJB) was created all fall into this category. Existing message include words like was created, about to create, and now exists, for example:

- “New log file was created”

DependencySituation

This category deals with the situations where components cannot find some component or feature that they require. This category includes messages about not finding the ”version” of the component that was expected. Messages that say a resource was not found, or that an application or subsystem that was unavailable, also fall into this category. Existing messages include words like “could not find”, and “no such component”, for example:

- “Error encountered while deploying database schema: no database found”

ConnectSituation

This category deals with the situations related to aspects about a connection attempt from one component to another component. Messages that say a connection failed, that a connection was created, or that a connection was ended all fall into this category. Existing messages include words like “connection reset”, “connection failed”, and “failed to get a connection”, for example:

- “Connection creation failed”
- “Connection with http://foo.com created”
- “Failed to close a connection”

ReportSituation

This category deals with situations that occur as a result of some setting or occurrence that causes the resource to asynchronously report various types of data. Types of information that falls into this category are:

- **Exception related** – some exception has occurred within the resource and it not covered by any other category.
- **Performance related** – some event occurs, that does not fall into any other category, that has affected performance in some way. For example, weather conditions may be affected line quality and network speeds are affected.

- **Security related** – some security issue has been detected, like the cabinet door to a secure piece of equipment has been opened or an attack of some sort has been detected.
- **Heartbeat related** – the resource has been configured to periodically report a ‘heartbeat’.
- **Status related** – some change of status that does not affect availability or capability of the resource has been detected. For example, printer ink cartridge is low.
- **Log related** – the resource has been configured to generate a log entry based on some event or at a fixed interval. This category identifies this event as a requested log entry.
- **Debug related** – the resource has been enabled to turn on diagnostic information flow and will report the information within this category.
- **Trace related** – the resource has been enabled to run trace information and reports this information using this category

OtherSituation

This category is for those events that do not fall into any other category. Note that this category is defined for syntactic completeness but any events placed in this category will not be able to be effectively correlated and its use is therefore discouraged unless absolutely necessary.