



Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.1

OASIS Standard, 01 August 2006

Document identifier:

wsdm-mows-1.1-spec-os-01

Location:

<http://docs.oasis-open.org/wsdm/wsdm-mows-1.1-spec-os-01.pdf>

Technical Committee:

OASIS Web Services Distributed Management TC

Chair(s):

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

Editors:

Kirk Wilson, Computer Associates kirk.wilson@ca.com

Igor Sedukhin, Computer Associates.

Abstract:

The Web Services Distributed Management (WSDM) specifications, as declared in the committee charter, define A) how management of any resource can be accessed via Web services protocols – Management Using Web Services, or MUWS, and B) management of the Web services resources via the former – Management Of Web Services, or MOWS. This document is the WSDM specification defining MOWS.

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).

36
37
38

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

Table of Contents

40	1	Introduction	5
41	1.1	Terminology.....	5
42	1.2	Notational conventions.....	6
43	2	Architecture.....	7
44	2.1	In-band and Out-of-band Manageability	8
45	2.2	Application to Resources Exposed as Web Services	8
46	2.3	Self-Management.....	9
47	3	Managing Web Services	10
48	3.1	Responsibilities of the Implementations of the Manageability Endpoints	10
49	3.2	Manageability at the Web service level.....	11
50	3.3	Using manageability of Web services endpoints	11
51	4	Security Considerations	13
52	4.1	Additional security considerations when managing Web services	13
53	5	Web service manageability capabilities	15
54	5.1	Common manageability capabilities.....	15
55	5.1.1	Manageability References	16
56	5.1.1.1	Operations.....	17
57	5.1.1.1.1	GetManageabilityReferences.....	17
58	5.2	Web service endpoint manageability capabilities	17
59	5.2.1	Identity	17
60	5.2.2	Identification.....	18
61	5.2.2.1	Properties	18
62	5.2.2.2	Events.....	19
63	5.2.3	Metrics	19
64	5.2.3.1	Information markup declarations	20
65	5.2.3.2	Properties	20
66	5.2.3.3	Events.....	23
67	5.2.4	Operation Metrics.....	23
68	5.2.4.1	Properties	24
69	5.2.4.2	Events.....	25
70	5.2.5	Operational State	25
71	5.2.5.1	Information markup declarations	25
72	5.2.5.2	Properties	27
73	5.2.5.3	Events.....	27
74	5.2.6	Operational Status	28
75	5.2.6.1	Events.....	28
76	5.2.7	Operation Operational Status	29
77	5.2.7.1	Properties	29
78	5.2.7.2	Events.....	30
79	5.2.8	Request Processing State	30
80	5.2.8.1	Information markup declarations	31
81	5.2.8.2	Events.....	32

82	5.2.8.2.1	RequestProcessingNotification message.....	36
83	5.2.8.2.2	Examples of events against the Web service endpoint request processing state...	38
84	6	References.....	41
85	6.1	Normative	41
86	6.2	Non-normative.....	41
87		Appendix A. Acknowledgments	43
88		Appendix B. Revision History	44
89		Appendix C. Notices	46
90		Appendix D. XML Schemas.....	47
91		Appendix E. WSDL elements	55
92		Appendix F. Notification topic spaces.....	56
93			

94 1 Introduction

95 Web services are an integral part of the IT landscape, and, as such, are vital resources to many
96 organizations. Web services may interact with other Web services and are used in business
97 processes. Interacting Web services form a logical network which may span enterprise
98 boundaries. Managing such a logical network is critical for organizations that use Web services to
99 automate and integrate various internal functions, and deal with partners and clients
100 electronically. To manage the Web services network, one needs to manage the components that
101 form the network – the Web services endpoints. This part of the WSDM specification addresses
102 management of the Web services endpoints using Web services protocols **[MOWS-Reqs]**.

103

104 The *Management Of Web Services* (MOWS) specification is based on the concepts and
105 definitions expressed in the *Management Using Web Services* specification (MUWS) **[MUWS]**. It
106 is recommended that the reader is aware of the MUWS specification contents.

107

108 Definitions and examples in this document are based on the following specifications. It is
109 recommended that the reader is aware of their contents.

- 110 ▪ WS Architecture **[WS-Arch]**
- 111 ▪ XML **[XML]**
- 112 ▪ XML Namespaces **[XNS]**
- 113 ▪ XML Schema **[XMLS]**
- 114 ▪ SOAP **[SOAP]**
- 115 ▪ WSDL **[WSDL]**
- 116 ▪ WS-Addressing **[WS-A]**
- 117 ▪ WS-ResourceProperties **[WS-RP]**
- 118 ▪ WS-BaseNotification **[WS-N]**
- 119 ▪ WS-Topics **[WS-T]**
- 120 ▪ XML Path Language **[XPath]**

121

122 Section 5 and appendices D, E and F are *normative* specifications. The rest of the document is
123 *non-normative*, and is provided as a background and explanatory material.

124

125 1.1 Terminology

126 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",
127 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be
128 interpreted as described in **[RFC2119]**.

129

130 This specification is based on the terminology defined in the WSDM **[MUWS]** specifications. In
131 addition, the following terms are defined.

132 ***Manageable Web service endpoint*** – is a Web service endpoint as a manageable resource.

133

134 1.2 Notational conventions

135 This specification uses an informal syntax to describe the XML grammar of the messages,
136 property instances and event information forming the manageability capability interfaces. This
137 syntax uses the following rules:

- 138 ▪ The syntax appears as an XML instance, but the values indicate the data types instead of
139 values.
- 140 ▪ {any} is a placeholder for elements from some other namespace (like ##other in XML
141 Schema).
- 142 ▪ Characters are appended to attributes, elements, and {any} to indicate the number of
143 times they may occur as follows: ? (0 or 1), * (0 or more), + (1 or more). No character
144 indicates exactly 1 occurrence. The characters [and] are used to indicate that contained
145 items are to be treated as a group with respect to the ?, *, and + characters.
- 146 ▪ Attributes, elements, and values separated by | and grouped with (and) are meant to be
147 syntactic alternatives.
- 148 ▪ ... is used in XML start elements to indicate that attributes from some other namespace
149 are allowed.
- 150 ▪ The XML namespace prefixes are used to indicate the namespace of the element being
151 defined

152 A full WSDL description of all interfaces and XML Schemas of all information elements are
153 available in the appendices.

154

155 When describing instances of XML information, and in order to refer to elements and attributes,
156 this specification uses a simplified XPath [**XPath**] notation which can be formally defined as
157 follows.

- 158 ▪ Path = '/'? (['@' ? (NCName | QName | "*")] | ['(' (NCName | QName | "*") ') '] '/' Path) ?
- 159 ▪ NCName is an XML non-qualified name as defined by XML Schema [XMLS]. In this case
160 the namespace is assumed to default to the namespace of this specification.
- 161 ▪ QName is an XML qualified name as defined by XML Schema [XMLS].
- 162 ▪ The symbol * denotes any name match.
- 163 ▪ The symbol / denotes a path delimiter. If it appears as the first element of the path, it
164 denotes the root of the XML document.
- 165 ▪ The symbol @ denotes a reference to an XML attribute, otherwise NCName, QName or *
166 refer to an XML element.
- 167 ▪ The symbols (and) denote a reference to an XML Schema type.

168

169 For example, /E1/E2/@A1 refers to an attribute A1 of an element E2 contained in element E1
170 which is a root of the XML document. E1/ns1:E2/E3 refers to an element E3 which is contained in
171 the element E2 which is contained in the element E1 anywhere in the XML document. In this case
172 element E2 belongs to the namespace mapped to the prefix ns1. (ns2:T1)/E1/ns1:E2/@A1 refers
173 to an attribute A1 on an element E2 contained in the element E1 declared in the XML Schema
174 type T1 which target namespace is mapped to the prefix ns2.

175

176

2 Architecture

Management of Web services (MOWS) is an application of Management using Web services (MUWS) to the resources that are elements of the Web Services Architecture [WS-Arch]. This WSDM specification defines how the manageability of Web service endpoints and resources exposed as Web services can be accessed via Web services. In order to achieve this goal, MOWS is based on the MUWS specifications, and the architecture, definitions and dependencies thereof [MUWS].

Application of the WSDM architecture concepts (section 2 of the MUWS specification part 1) to the management of Web services could be described as follows (Figure 1). A *manageability Web service endpoint* (or, shortly, *manageability endpoint*) provides access to the *manageable Web service endpoint resource* (a manageable resource, in terms of MUWS). A manageable Web service endpoint (or, shortly, *manageable endpoint*) could be, for example, an endpoint of an order entry Web service for which received messages could be counted and reported to the *manageability consumers*. Following the WSDM concepts, the manageability consumer discovers the manageability endpoint and exchanges messages with it in order to request information, subscribe to events or control the manageable endpoint resource.

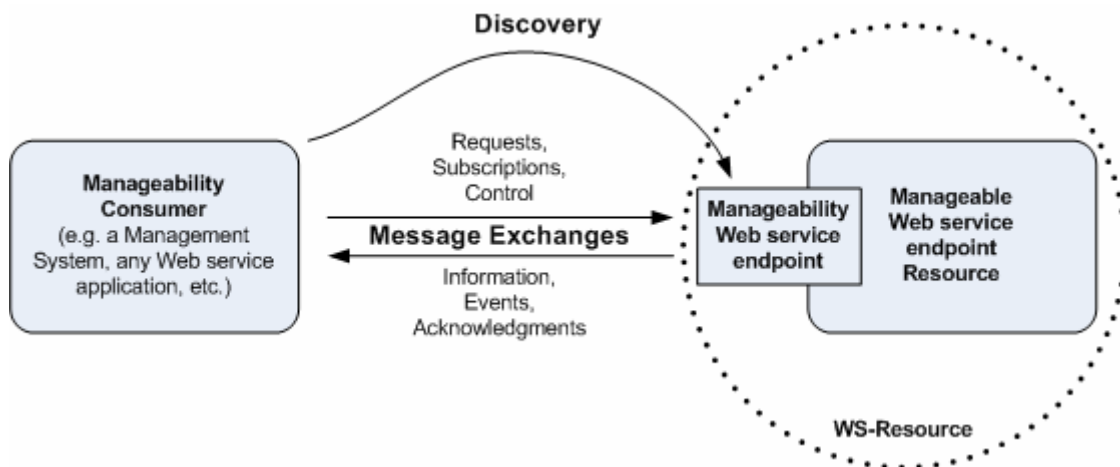


Figure 1. Management of Web services concepts

Refer to section 2 of the MUWS specification part 1 [MUWS] for more detailed explanation of discovery and message exchange between manageability consumers and manageability endpoints.

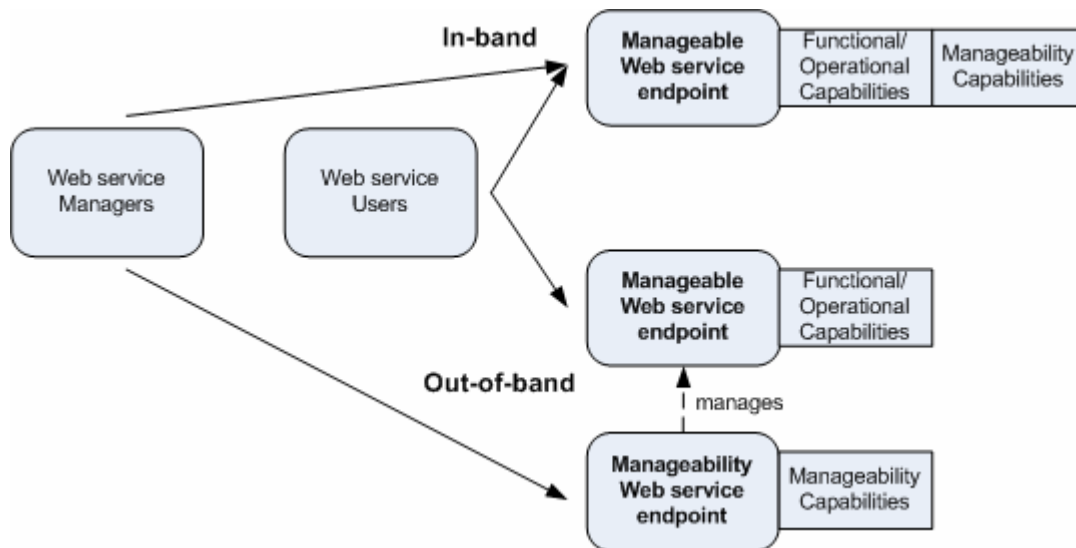
The following are important aspects of the WSDM architecture.. Please refer to the referenced sections of the MUWS specification [MUWS]

- **Focus on resources** (section 2.1 of MUWS part 1) – focus on providing access to the manageable resources – a contract between a manageability consumer and a manageable resource with regards to discovery and message exchanges.
- **Composeability** (section 2.2 of MUWS part 1) – allows a non-conflicting, incremental mix of Web services implementation aspects and manageability capabilities.

209
210
211
212
213
214
215
216
217

2.1 In-band and Out-of-band Manageability

A unique feature of the MOWS subject domain is that a manageability endpoint and a manageable endpoint are both Web services endpoints, and therefore could be the same endpoint or could be different endpoints. In other words, manageability consumers and regular Web service consumers could target their messages to the same or to different endpoints. Either of the approaches is allowed by the MOWS architecture and the implementation choices are transparent for manageability consumers (and Web service consumers, for that matter). The Figure 2 illustrates this.



218
219
220

Figure 2. In-band and out-of-band manageability

221

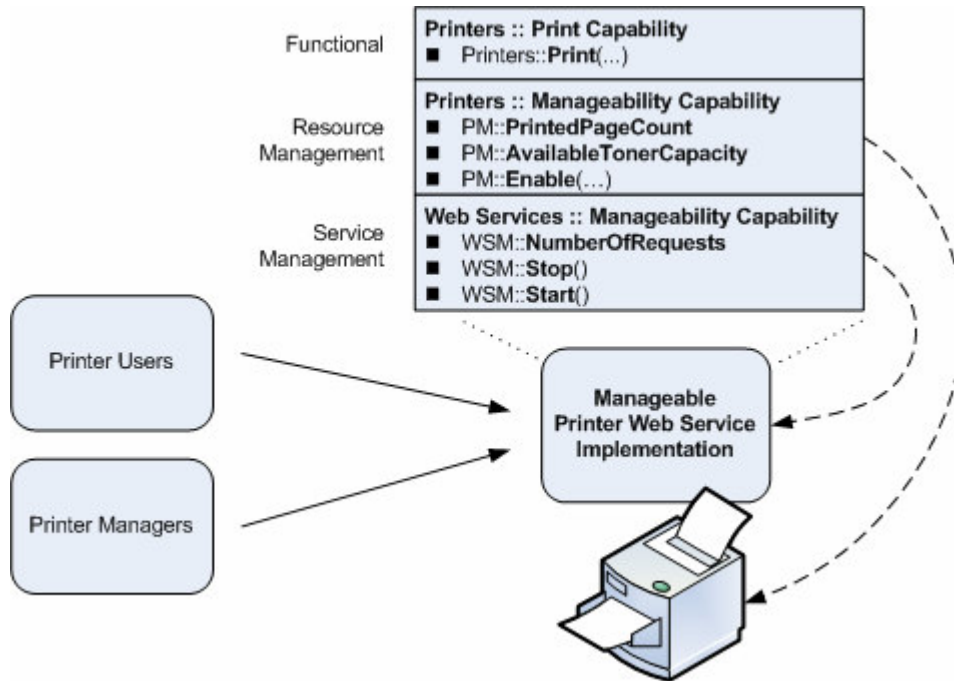
2.2 Application to Resources Exposed as Web Services

222 WSDM allows a resource and all of its services to be manageable in a standard and interoperable
223 manner. A resource may support both manageability and functional capabilities. For example, a
224 printer can obviously print, but the same printer may also be able to indicate if it is on-line and
225 may be able to notify when the toner is running out. A manageable resource may allow access to
226 its manageability capabilities and functional capabilities via Web services. Web services
227 represent a composition of manageable and functional qualities of a given resource (Figure 3).

228 Manageability consumers might take advantage of a composition of manageability and functional
229 capabilities: 1) management-oriented consumers gain visibility into functional aspects of a
230 resource 2) business-oriented consumers gain visibility into management aspects of a resource.
231 For example, a Web services-based business process may involve a selection of an on-line
232 printer with sufficient amount of toner in order to print an urgent report for executives.

233 Composeability makes it easy for implementers of resource services to offer an appropriate set of
234 functional capabilities along with an appropriate set of manageability capabilities guided by the
235 appropriate model for authorization of these requests.

236

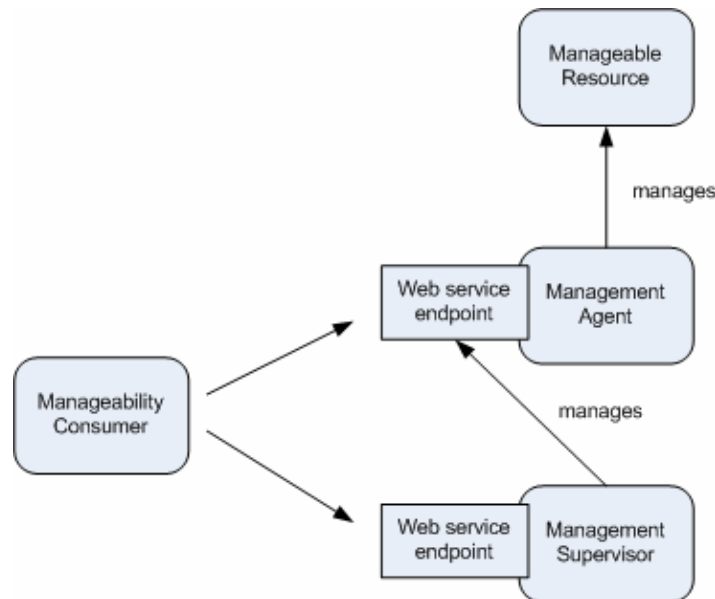


237
238

Figure 3. Application to resources exposed as Web services

239 **2.3 Self-Management**

240 The WSDM specifications define how to use Web services to expose manageable resources
 241 (MUWS), and in addition, define how to expose manageable Web service implementations
 242 (MOWS – this specification). Application of MOWS to MUWS gives an interesting combination of
 243 the manageable management. Using both specifications, it is possible to build reliable and
 244 accountable management systems (Figure 4).



245
246
247

Figure 4. Applying MOWS to MUWS

248 3 Managing Web Services

249 Using definitions expressed in WSDL 1.1 [WSDL] and WS-Addressing [WS-A] as guidelines, a
250 Web service (described by a WSDL 1.1 service element) is an aggregate of endpoints (described
251 by WSDL 1.1 port elements). An endpoint binds a Web service interface (described by a WSDL
252 1.1 portType element) to an address (URI). Each interface describes a set of messages that
253 could be exchanged and their format. Properly formatted messages could be sent to the endpoint
254 at the address in the way prescribed by the binding (described by a WSDL 1.1 binding element).
255 A Web service description contains definitions of a combination of interfaces and services.

256

257 According to the section 2, management of Web services starts at an endpoint resource which,
258 therefore, becomes a manageable resource, specifically called a manageable endpoint. The
259 reason the Web service endpoint is the basic manageable resource is that (1) anything behind an
260 endpoint is a concrete implementation (e.g. an application hosted on a server), and (2) an
261 aggregate of endpoints is a logical construct, management of which has to be inferred from
262 manageability of the constituent endpoints. This specification focuses on defining manageability
263 capabilities of the Web service endpoints. Furthermore, (1) is in the realm of the
264 applications/systems/networks management, and (2) should be done by the intelligent
265 management systems. Aspects of (1) are further discussed in section 3.1. Aspects of (2) are
266 further discussed in section 3.2.

267

268 This specification balances requirements of Web services management applications and the
269 complexity of implementing manageability endpoints.

270 3.1 Responsibilities of the Implementations of the Manageability 271 Endpoints

272 The system providing manageability capabilities for a Web service endpoint must be aware of the
273 environment as experienced from the Web service caller's point of view. This *experience* may be
274 dependent upon hardware or software configuration in which the Web service endpoint exists.
275 Implementations of manageability endpoints may need to account for management requests
276 made with respect to the Web service caller's point of view.

277

278 Consider two examples. The first case is that of a hardware routing configuration. A hardware
279 device controls access to all messages sent to a particular URL such as
280 <http://external.example.com/theService>. Upon receipt of messages for that URL, the device
281 distributes the messages to Web service endpoints at the <http://s1.example.com/theService>,
282 <http://s1.example.com/theService>, and <http://s2.example.com/theService> addresses.

283

284 If, say, a query regarding metrics were made regarding the Web service endpoint receiving
285 messages at the <http://external.example.com/theService> address, it is the responsibility of the
286 implementation of the manageability endpoint to aggregate the results from the three underlying
287 Web service endpoints to provide a meaningful response.

288

289 A second example is one in which a single Web service endpoint is accessible at two distinct
290 URLs due to DNS aliasing. Consider the Web service endpoint at
291 <http://services.example.com/creditCheck>. External to the Example Company, this Web service
292 endpoint is accessible at the <http://ourservices.example.com/creditCheck> address, while
293 internally, this Web service endpoint is accessible at <http://extservices.example.com/creditCheck>.

294 However, in both cases, the message processing is performed by the same machine, application,
295 code, etc. The Web service endpoint implementation itself is *aware* of the means by which it is
296 addressed (e.g. is using the URL header of the HTTP messages), and it adjusts message
297 processing appropriately.

298

299 In this case, the implementation of the manageability endpoint must be similarly aware of how the
300 Web service endpoint was accessed. Queries regarding the two URL aliases must be accounted
301 for separately, even though the underlying Web service endpoint is the same.

302 **3.2 Manageability at the Web service level**

303 Management applications may want to manage Web services at the granularity level of the
304 endpoint. For example, to find out when an endpoint goes down and how many messages a
305 specific endpoint has processed. At the same time, there are many cases where the
306 management applications may want to manage the Web service as a logical aggregate of all of
307 its endpoints. For example, a business manager using a business dashboard doesn't care
308 whether the purchase orders arrive via the HTTP or the SMTP binding of the order entry Web
309 service, or whether orders arrive via the US server or its European mirror.

310

311 In recognition of these requirements, this specification defines manageability of endpoints as the
312 base building block for managing Web services. The specification ensures that information is
313 available to management applications in order to summarize to the Web service-level view. This
314 includes allowing manageable endpoints to establish relationships linking them as part of the
315 same Web service.

316 **3.3 Using manageability of Web services endpoints**

317 The following pattern may be used by the manageability consumers which intend to manage Web
318 services endpoints.

- 319 1. Obtain an EPR to the manageability endpoint. One of the following ways may be used.
 - 320 a. Discover manageable resources as described in the MUWS specifications
321 **[MUWS]**.
 - 322 b. Exercise the Manageability References capability (section 5.1.1) on the
323 functional Web services endpoint.
 - 324 c. The functional Web services endpoint may also be the manageability endpoint
325 (section 2.1). Determine that by detecting if the endpoint supports the MUWS
326 Identity capability:
 - 327 i. Either, obtain the WSDL document describing the manageability
328 endpoint and look for a ResourceId element (see MUWS specification
329 part 1 section 5.1) in the first level children of the resource properties
330 document root **[WS-RP]**.
 - 331 ii. Or, request the value of the ManageabilityCapability property (see
332 MUWS specification part 1 section 5.2) and look for the URI which
333 identifies the MUWS Identity capability.
- 334 2. Using the EPR obtained in the previous step, and based on the manageability capabilities
335 intended to be used, build Web services messages targeted at the manageable Web
336 services endpoint.
 - 337 a. Obtain the WSDL document describing the manageability endpoint and
338 understand how operations defined by the manageability capabilities are bound.

339
340
341

b. Request the value of the ManageabilityCapability property (see MUWS specification part 1 section 5.2) and look for the URIs which identify the capabilities to be used.

342
343
344
345

c. To understand how to construct Web services messages for management of a Web services endpoint, consult the manageability capability definition sections in this specification or in the MUWS specification and any dependent specifications thereof.

346 4 Security Considerations

347 It is RECOMMENDED that communication between a manageability consumer and a
348 manageability endpoint be secured using the mechanisms described in WS-Security [WSS] and
349 WS-I Basic Security Profile [BSP], including transport-level security such as HTTP over Secure
350 Socket Layers (SSL). In order to properly secure messages, the body and all relevant headers
351 may need to be signed and encrypted.

352 The following list summarizes common classes of attacks that apply generally to protocols and
353 identifies mechanisms available to prevent/mitigate the attacks:

- 354 ▪ **Message alteration** – Alteration is prevented by including signatures of the message
355 information using WS-Security.
- 356 ▪ **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using
357 WS-Security.
- 358 ▪ **Key integrity** – Key integrity is maintained by using the strongest algorithms possible.
- 359 ▪ **Authentication** – Authentication is established using the mechanisms described in WS-
360 Security and other related specifications. Each message is authenticated using the
361 mechanisms described in WS-Security.
- 362 ▪ **Accountability** – Accountability is a function of the type of and strength of the key and
363 algorithms being used. In many cases, a strong symmetric key provides sufficient
364 accountability. However, in some environments, strong PKI signatures are required.
- 365 ▪ **Availability** – All services are subject to a variety of availability attacks. Replay detection
366 is a common attack and it is RECOMMENDED that this be addressed by the
367 mechanisms described in WS-Security. Other attacks, such as network-level denial of
368 service attacks are harder to avoid and are outside the scope of this specification. That
369 said, care should be taken to ensure that minimal state is saved prior to any
370 authenticating sequences.

371

372 The WS-I Basic Security Profile working group has produced a scenarios document which
373 explores these threats in more detail and which identifies security requirements which are then
374 addressed by subsequent profiles [BSP]. WSDM looks to the security domain experts to define
375 the mechanisms to secure web services and looks to WS-I to define interoperability profiles that
376 can be leveraged by WSDM implementers.

377

378 4.1 Additional security considerations when managing Web 379 services

380 It is RECOMMENDED that the implementers of manageability endpoints and manageability
381 consumers take into consideration the following security related concerns.

- 382 ▪ If a manageable Web services endpoint supports messages from both a consumer of a
383 service and a manager of a service section 2.1, it may be important to identify a security
384 model which allows for the appropriate level of granularity with regard to the message
385 origin. For example, setting configuration options may be allowed by a manageability
386 consumer but not an application consumer. When these composed services are
387 deployed, it will be important to understand the authorization model for both management
388 and functional use.
- 389 ▪ In order to make the management systems secure in addition to reliable and accountable
390 (section 2.3), it will be important to follow a set of guidelines and best practices that detail

391 how to compose MOWS with existing security implementations and emerging
392 specifications for authorization and trust.

393 ■ Implementers of this specification may need to give a particular attention to security when
394 implementing the following manageability capabilities.

395 ○ Manageability References (section 5.1.1) – this capability allows access to the
396 manageability endpoint references of a functional Web service endpoint. The
397 concern is that visibility to these references may need to be protected differently
398 than visibility of the functional Web service endpoint and its operations.

399 ○ Request Processing State (section 5.2.8) – this capability allows managers to
400 subscribe to notifications against request processing by a functional Web service
401 endpoint.

402 1. Not all managers should be allowed to subscribe to request processing
403 notification because messages may contain protected information, and/or
404 may be used to generate a DoS attack.

405 2. The request messages may be encrypted and signed. Therefore, managers
406 may need to possess information that allows them to deal with such
407 encrypted and signed messages.

408 3. Notification messages which contain information about request messages
409 SHOULD be encrypted to avoid spoofing of this information by intercepting
410 notification messages.

411 4. The request processing notification message provides sufficient flexibility
412 with respect to its content to avoid inclusion of information which needs to be
413 highly protected and therefore not relayed to managers.

414

5 Web service manageability capabilities

415

416 The following sections define manageability capabilities for Web services and resources exposed
417 as Web services (see 2.2).

418

419 Each capability is described in a UML summary diagram. Metadata is defined for properties,
420 operations and events according to MUWS specification part 1 section 3.4 and part 2 section 2.4
421 **[MUWS]**.

422

423 The definitions of the Web service manageability capabilities are rendered into WSDL elements
424 (interfaces/portTypes) and supporting XML Schemas in Appendix D, and Appendix E, and
425 Appendix F contains renditions of the notification topic spaces for the events defined by the
426 capability specifications.

427

428 Following namespace prefixes are used in this document when referring to XML elements and
429 XML schemas. The table below describes what prefix corresponds to which namespace URI.

430

Prefix	Namespace
muws1	http://docs.oasis-open.org/wsdm/muws1-2.xsd
muws2	http://docs.oasis-open.org/wsdm/muws2-2.xsd
mows	http://docs.oasis-open.org/wsdm/mows-2.xsd
mowsw	http://docs.oasis-open.org/wsdm/mows-2.wsdl
mowse	http://docs.oasis-open.org/wsdm/mowse-2.xml
wsa	http://www.w3.org/2005/08/addressing
wsdl	http://www.w3.org/2002/07/wsdl
S	http://schemas.xmlsoap.org/soap/envelope/ or http://www.w3.org/2002/12/soap-envelope
xs	http://www.w3.org/2001/XMLSchema
wsrf-rp	http://docs.oasis-open.org/wsrf/rp-2
wstop	http://docs.oasis-open.org/wsn/t-1

431

432 Unless otherwise specified, XML elements and XML schema types introduced in this specification
433 belong to the namespace mapped to the **mows** prefix.

434

5.1 Common manageability capabilities

435

436 The following sections define manageability capabilities applicable to Web services and
437 resources exposed as Web services.

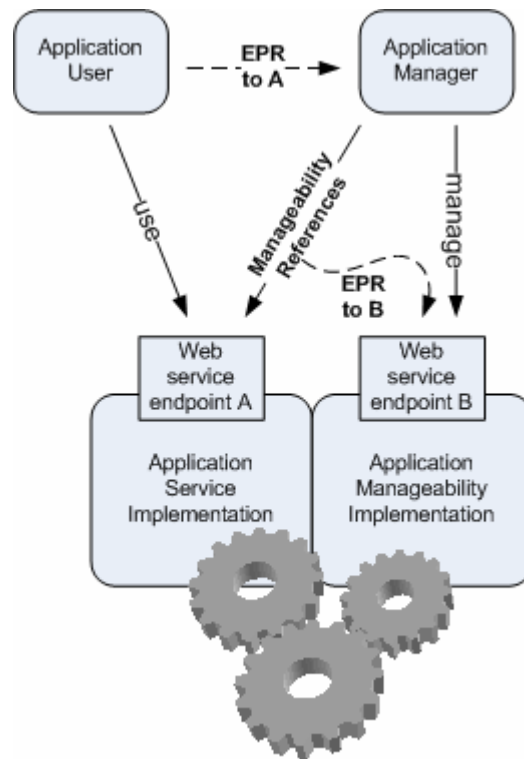
438 **5.1.1 Manageability References**

439 This capability is identified by the following URI:
440 <http://docs.oasis-open.org/mows2/capabilities/ManageabilityReferences>

441
442 This capability allows a functional/operational Web service or a resource exposed as a Web
443 service (section 2.2) (*the service*) to provide references to its manageability endpoints. This
444 capability is intended for implementations of functional/operational Web services endpoints. The
445 consumer may exchange messages with *the service* in order to request references to the
446 manageability endpoints. Using obtained references, the consumer may exchange messages
447 with the manageability endpoints in order to perform management activities to *the service*.

448
449 For example (Figure 5), an application user accesses a Web service endpoint A. The application
450 user then gives the endpoint A reference to the application manager which accesses the Web
451 service endpoint A in order to obtain a reference to the application manageability implementation
452 accessible at the Web service endpoint B. The application manager may now manage the
453 application by exchanging management related messages with endpoint B.

454



455

456 **Figure 5.** Use of Manageability References capability

457

458 The name of this capability identifies its semantics of providing references to manageability
459 endpoints of the service that supports this capability. Within this specification, this capability
460 consists of one operation:

- 461 • GetManageabilityReferences.

462 but applications may associate further operations with this capability.

463 **5.1.1.1 Operations**

464 The following is the specification of the Manageability References capability operations.

465

466 **5.1.1.1.1 GetManageabilityReferences**

467 This operation is mandatory for implementations of this capability and is defined as the following
468 message exchange.

469

470 The request to perform this operation is a message containing the following XML element.

471

```
472 <GetManageabilityReferences/>
```

473 **GetManageabilityReferences** is a Global Element Declaration (GED) which identifies the
474 request of the GetManageabilityReferences operation.

475 The wsa:Action MUST contain the URI

476 <http://docs.oasis-open.org/wsdm/mows/GetManageabilityReferences>.

477

478 The response to the above request is either a fault (any fault) or a message containing the
479 following XML element.

480

```
481 <GetManageabilityReferencesResponse>  
482 <muws1:ManageabilityEndpointReference>  
483 <!-- see [MUWS] -->  
484 </muws1:ManageabilityEndpointReference>+  
485 </GetManageabilityReferencesResponse>
```

486 The wsa:Action MUST contain the URI

487 <http://docs.oasis-open.org/wsdm/mows/GetManageabilityReferencesResponse>.

488

489 **GetManageabilityReferencesResponse** is a GED which identifies the response to the
490 requested GetManageabilityReferences operation.

491

492 **GetManageabilityReferencesResponse/muws1:ManageabilityEndpointReference** is a
493 reference to the Web service endpoint which provides access to the management of the
494 functional/operational Web service endpoint or the Web service-enabled resource which
495 responded to the GetManageabilityReferences operation request message.

496

497 **5.2 Web service endpoint manageability capabilities**

498 The following sections define manageability capabilities applicable to Web service endpoints.

499 **5.2.1 Identity**

500 A WSDM manageable endpoint MUST support the MUWS **Identity** manageability capability
501 (section 5.1 of the [MUWS] part 1). There are no extensions to the MUWS definition of this
502 capability.

503 5.2.2 Identification

504 This capability is identified by the following URI:

505 **http://docs.oasis-open.org/mows-2/capabilities/Identification**

506 All properties, operations and events defined for this capability have the following metadata:

- 507 ▪ `<muws2:Capability>http://docs.oasis-open.org/mows-`
508 `2/capabilities/Identification</muws2:Capability>`

509

510 The Web service endpoint's manageable identification capability is represented by the
511 **Identification** UML model class. The name of this capability identifies its semantics. This
512 capability name and semantics are consistent with the following definition (from the Webster
513 dictionary).

514 identification: **1 a** : an act of identifying : the state of being identified **b** : evidence of
515 identity

516

517 Note that, in contrast, the MUWS **Identity** capability and semantics are consistent with the
518 following definition (from the Webster dictionary).

519 identity: **1 a** : sameness of essential or generic character in different instances **b** :
520 sameness in all that constitutes the objective reality of a thing : ONENESS

521

522 The *identification* capability is used to help establish the Web service endpoint being managed.
523 The *identity* capability may be used to determine if two manageability endpoints provide
524 manageability of the same resource or not.

525 This capability consists of two properties:

- 526 • EndpointReference (mandatory)
527 • EndpointDescriptions (optional)
528 • Description (zero to many)

529 and defines one event: IdentificationCapability,

530 but applications may associate further properties and events with this capability.

531 5.2.2.1 Properties

532 The following is the specification of the Web service endpoint identification properties (i.e. XML
533 elements which represent properties).

534

```
535 <EndpointReference>wsa:EndpointReferenceType</EndpointReference>  
536 <EndpointDescriptions><description>xs:anyURI</description>*</EndpointDescriptions>?
```

537

538 **EndpointReference** is a reference to the Web service endpoint being managed. A reference
539 must be resolvable to the actual useable endpoint. This property represents one way to access
540 the endpoint resource but doesn't preclude the existence of multiple descriptions of the same
541 endpoint resource. Metadata about this property is as follows.

- 542 ▪ Is not *Mutable*
543 ▪ Is not *Modifiable*

544 **EndpointDescriptions** is a list of URIs pointing to description documents of the Web service
545 endpoint resource. The different description documents can be of the same or of different types
546 (e.g. WSDL1.1, WSDL2.0, UDDI tModel, etc.). Metadata about this property is as follows.

- 547 ▪ Is *Mutable*
- 548 ▪ Is not *Modifiable*

549 **5.2.2.2 Events**

550 The following specification defines this capability notification topics in the namespace mapped to
551 the **mowse** prefix.

552

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

554

555 **mowse:IdentificationCapability** is a topic on which management events related to this
556 manageability capability SHOULD be emitted.

557

558 Property change events MUST be wrapped in Management events and published on topics
559 defined in this section.

560

561 **5.2.3 Metrics**

562 This capability is identified by the following URI:

563 **<http://docs.oasis-open.org/mows-2/capabilities/Metrics>**

564 All properties, operations and events defined for this capability have the following metadata:

- 565 ▪ <muws2:Capability>[http://docs.oasis-open.org/mows-](http://docs.oasis-open.org/mows-2/capabilities/Metrics)
566 2/capabilities/Metrics</muws2:Capability>

567

568 The name of this capability identifies its semantics of providing properties that are useful in
569 measuring the use and performance of Web services. This capability consists of ten properties:

- 570 • NumberOfRequests (optional)
- 571 • NumberOfFailedRequests (optional)
- 572 • NumberOfSuccessfulRequests (optional)
- 573 • ServiceTime (optional)
- 574 • MaxResponseTime (optional)
- 575 • LastResponseTime (optional)
- 576 • MaxRequestSize (optional)
- 577 • LastRequestSize (optional)
- 578 • MaxResponseSize (optional)
- 579 • LastResponseSize (optional)

580 and defines one event: MetricsCapability,

581 but applications may associate further properties and events with this capability.

582

583 This capability extends the definition of the MUWS Metrics capability. WSDM manageable
584 endpoints that intend to support the MOWS **Metrics** capability MUST support the MUWS **Metrics**
585 capability (section 3.4 of the **[MUWS]** part 2) as well.

586

587 It is recommended that for adequate calculations, the Web service endpoint metric properties
588 (one or all) are retrieved together with the **muws2:CurrentTime** property (e.g., using one request
589 to retrieve multiple properties).

590

591 Metrics and request processing states are related. The request processing state change
592 boundaries are the points where metric counters are incremented. These states are defined
593 below, in section 5.2.8.

594 **5.2.3.1 Information markup declarations**

595 The following two XML Schema complex types are defined for metrics that represent integers and
596 durations of time.

597

```
598 <xs:complexType name="IntegerCounter">  
599   <xs:simpleContent>  
600     <xs:extension base="xs:nonNegativeInteger">  
601       <xs:attributeGroup ref="muws2:MetricAttributes"/>  
602       <xs:anyAttribute namespace="##other" processContents="lax"/>  
603     </xs:extension>  
604   </xs:simpleContent>  
605 </xs:complexType>
```

606

607 **(IntegerCounter)** type declares an xs:nonNegativeInteger counter metric.

608

```
609 <xs:complexType name="DurationMetric">  
610   <xs:simpleContent>  
611     <xs:extension base="xs:duration">  
612       <xs:attributeGroup ref="muws2:MetricAttributes"/>  
613       <xs:anyAttribute namespace="##other" processContents="lax"/>  
614     </xs:extension>  
615   </xs:simpleContent>  
616 </xs:complexType>
```

617

618 **(DurationMetric)** type declares an xs:duration metric.

619 **5.2.3.2 Properties**

620 The following is the specification of the Web service endpoint metrics properties (i.e. XML
621 elements which represent properties).

622

```
623 <NumberOfRequests>IntegerCounter</NumberOfRequests>?  
624 <NumberOfFailedRequests>IntegerCounter</NumberOfFailedRequests>?  
625 <NumberOfSuccessfulRequests>IntegerCounter</NumberOfSuccessfulRequests>?  
626 <ServiceTime>DurationMetric</ServiceTime>?  
627 <MaxResponseTime>DurationMetric</MaxResponseTime>?  
628 <LastResponseTime>DurationMetric</LastResponseTime>?  
629 <MaxRequestSize>IntegerCounter</MaxRequestSize>?  
630 <LastRequestSize>IntegerCounter</LastRequestSize>?  
631 <MaxResponseSize>IntegerCounter</MaxResponseSize>?  
632 <LastResponseSize>IntegerCounter</LastResponseSize>?
```

633

634 **NumberOfRequests** is a counter of the number of request messages that the Web service
635 endpoint has received. This counter is incremented by 1 whenever a request reaches the
636 Received state according to the Request Processing Model (Figure 6). Metadata about this
637 property is as follows.

- 638 ▪ Is *Mutable*
- 639 ▪ Is not *Modifiable*
- 640 ▪ <muws2:ChangeType>Counter</muws2:ChangeType>
- 641 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 642 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 643 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

644 **NumberOfFailedRequests** is a counter of the number of request messages that the Web service
645 endpoint has received, and a (SOAP) fault was sent in reply. This counter is incremented by 1
646 whenever a request reaches the Failed state according to the Request Processing Model (Figure
647 6). Metadata about this property is as follows.

- 648 ▪ Is *Mutable*
- 649 ▪ Is not *Modifiable*
- 650 ▪ <muws2:ChangeType>Counter</muws2:ChangeType>
- 651 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 652 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 653 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

654 **NumberOfSuccessfulRequests** is a counter of the number of request messages that the Web
655 service endpoint has received, and anything but a (SOAP) fault was sent in reply. This counter is
656 incremented by 1 whenever a request reaches the Completed state according to the Request
657 Processing Model (Figure 6). Metadata about this property is as follows.

- 658 ▪ Is *Mutable*
- 659 ▪ Is not *Modifiable*
- 660 ▪ <muws2:ChangeType>Counter</muws2:ChangeType>
- 661 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 662 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 663 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

664 Note that **NumberOfSuccessfulRequests + NumberOfFailedRequests ≤ NumberOfRequests**
665 as there could possibly be some requests that were received, but lost or still being processed.

666

667 **ServiceTime** is a counter of the total elapsed time that the Web service endpoint has taken to
668 process all requests (successfully or not). Metadata about this property is as follows.

- 669 ▪ Is *Mutable*
- 670 ▪ Is not *Modifiable*
- 671 ▪ <muws2:ChangeType>Counter</muws2:ChangeType>
- 672 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 673 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 674 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

675 **MaxResponseTime** is a gauge indicating the maximum time duration between all requests
676 received and their completion or failure. Metadata about this property is as follows.

- 677 ▪ Is *Mutable*
- 678 ▪ Is not *Modifiable*

- 679 ▪ <muws2:ChangeType>Gauge</muws2:ChangeType>
- 680 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 681 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 682 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

683 **LastResponseTime** is a gauge indicating the last recorded time duration between the last
 684 request received and its completion or failure. Metadata about this property is as follows.

- 685 ▪ Is *Mutable*
- 686 ▪ Is not *Modifiable*
- 687 ▪ <muws2:ChangeType>Gauge</muws2:ChangeType>
- 688 ▪ <muws2:TimeScope>PointInTime</muws2:TimeScope>
- 689 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

690 **MaxRequestSize** is a gauge indicating the maximum size in bytes for all requests received
 691 regardless of their completion or failure. Metadata about this property is as follows.

- 692 ▪ Is *Mutable*
- 693 ▪ Is not *Modifiable*
- 694 ▪ <muws2:Units>byte</muws:Units>
- 695 ▪ <muws2:ChangeType>Gauge</muws2:ChangeType>
- 696 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 697 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 698 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

699 **LastRequestSize** is a gauge indicating the last request size in bytes for the last request received
 700 regardless of the completion or failure of the request. Metadata about this property is as follows.

- 701 ▪ Is *Mutable*
- 702 ▪ Is not *Modifiable*
- 703 ▪ <muws2:Units>byte</muws:Units>
- 704 ▪ <muws2:ChangeType>Gauge</muws2:ChangeType>
- 705 ▪ <muws2:TimeScope>PointInTime</muws2:TimeScope>
- 706 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

707 **MaxResponseSize** is a gauge indicating the maximum response size in bytes for all responses
 708 sent regardless of their completion or failure. Metadata about this property is as follows.

- 709 ▪ Is *Mutable*
- 710 ▪ Is not *Modifiable*
- 711 ▪ <muws2:Units>byte</muws:Units>
- 712 ▪ <muws2:ChangeType>Gauge</muws2:ChangeType>
- 713 ▪ <muws2:TimeScope>SinceReset</muws2:TimeScope> or
- 714 ▪ <muws2:TimeScope>Interval</muws2:TimeScope>
- 715 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

716 **LastResponseSize** is a gauge indicating the size of the last response in bytes sent regardless of
 717 the completion or failure of the request. Metadata about this property is as follows.

- 718 ▪ Is *Mutable*
- 719 ▪ Is not *Modifiable*
- 720 ▪ <muws2:Units>byte</muws:Units>
- 721 ▪ <muws2:ChangeType>Gauge</muws2:ChangeType>

- 722 ▪ <muws2:TimeScope>PointInTime</muws2:TimeScope>
723 ▪ <muws2:GatheringTime>OnChange</muws2:GatheringTime>

724

725 Note that if a metric property has a <muws2:TimeScope>SinceReset</muws2:TimeScope>
726 metadata value, the muws2:ResetAt attribute MUST be reported on the property element and the
727 muws2:Duration attribute MUST NOT be reported. If a metric property has a
728 <muws2:TimeScope>Interval</muws2:TimeScope> metadata value, the muws2:ResetAt attribute
729 MAY be reported on the property element and the muws2:Duration attribute MUST be reported.

730

731 Also note that in this specification, counters are not just monotonically increasing variables, but
732 also represent a cumulative metric of some kind e.g. number of requests over time. Gauges, on
733 the other hand, do not represent a cumulative metric, but rather discrete values of some kind at a
734 particular time (e.g. response time).

735 5.2.3.3 Events

736 The following specification defines this capability notification topics in the namespace mapped to
737 the **mowse** prefix.

738

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

740

741 **mowse:MetricsCapability** is a topic on which management events related to this manageability
742 capability SHOULD be emitted.

743

744 Property change events MUST be wrapped in Management events and published on topics
745 defined in this section.

746

747 5.2.4 Operation Metrics

748 This capability is identified by the following URI:

749 **<http://docs.oasis-open.org/wsdm/mows-2/capabilities/OperationMetrics>**

750 All properties, operations and events defined for this capability have the following metadata:

- 751 ▪ <muws2:Capability>[http://docs.oasis-open.org/wsdm/mows-](http://docs.oasis-open.org/wsdm/mows-2/capabilities/OperationMetrics)
752 2/capabilities/OperationMetrics</muws2:Capability>

753

754 The name of this capability identifies its semantics of providing MOWS metrics at an operational
755 level. This capability applies MOWS metrics to operations within the managed Web service. It
756 consists of multiply occurring complex properties, each complex property represents an operation
757 of the service that may be characterized with MOWS metrics (see section 5.2.3.2):

- 758 • OperationMetrics (many)
- 759 • NumberOfRequests (optional)
 - 760 • NumberOfFailedRequests (optional)
 - 761 • NumberOfSuccessfulRequests (optional)
 - 762 • ServiceTime (optional)
 - 763 • MaxResponseTime (optional)
 - 764 • LastResponseTime (optional)

- 765 • MaxRequestSize (optional)
- 766 • LastRequestSize (optional)
- 767 • MaxResponseSize (optional)
- 768 • LastResponseSize (optional)

769 and defines one event: OperationMetricsCapability,
 770 but applications may associate further properties and events with this capability.
 771

772 WSDM manageable endpoints that intend to support the MOWS **Operation Metrics** capability
 773 MUST support the MUWS **Metrics** capability (section 3.4 of the [MUWS] part 2) as well. It is
 774 expected that support of the **Operation Metrics** capability will usually be done in addition to the
 775 MOWS **Metrics** capability, but this is not a requirement.

776
 777 It is recommended that for adequate calculations, the Web service operation metric properties
 778 (one or all) are retrieved together with the **muws2:CurrentTime** property.
 779

780 5.2.4.1 Properties

781 The following is the specification of the Web service endpoint operation metrics complex property
 782 (i.e. XML element that represents the metrics of an operation).

783

```

784 <OperationMetrics operationName="xs:NCName", portType="xs:QName"? ...>*
785   <NumberOfRequests>IntegerCounter</NumberOfRequests>?
786   <NumberOfFailedRequests>IntegerCounter</NumberOfFailedRequests>?
787   <NumberOfSuccessfulRequests>IntegerCounter</NumberOfSuccessfulRequests>?
788   <ServiceTime>DurationMetric</ServiceTime>?
789   <MaxResponseTime>DurationMetric</MaxResponseTime>?
790   <LastResponseTime>DurationMetric</LastResponseTime> ?
791   <MaxRequestSize>IntegerCounter</MaxRequestSize>?
792   <LastRequestSize>IntegerCounter</LastRequestSize>?
793   <MaxResponseSize>IntegerCounter</MaxResponseSize>?
794   <LastResponseSize>IntegerCounter</LastResponseSize>?
795   {any} *
796 </OperationMetrics>
  
```

797

798 **OperationMetrics** is the group property that contains the metrics for one operation within the
 799 managed Web service. The operation to which the metrics apply is identified by the attributes of
 800 this element.

801 **OperationMetrics/@operationName** is the name of operation to which the metrics apply. This is
 802 a required attribute. The value of this attribute is the name of the operation as it is specified in the
 803 portType of the Web service

804 **OperationMetrics/@portType** is the QName of the portType in which the operation occurs. This
 805 is an optional attribute. The value of this attribute is the QName of the portType as it is specified
 806 in the WSDL of the Web service.

807 The metrics contained within the OperationMetrics complex property are the MOWS metrics
 808 defined in section 5.2.3.2.

809

810 **5.2.4.2 Events**

811 The following specification defines the capability notification topics in the namespace mapped to
812 the **mowse** prefix.

813

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

815

816 **mowse:OperationMetricsCapability** is a topic on which management events related to this
817 manageability capability SHOULD be emitted.

818

819 **5.2.5 Operational State**

820 This capability is identified by the following URI:

821 **<http://docs.oasis-open.org/mows-2/capabilities/OperationalState>**

822 All properties, operations and events defined for this capability have the following metadata:

- 823 ▪ `<muws2:Capability>http://docs.oasis-open.org/mows-`
824 `2/capabilities/OperationalState</muws2:Capability>`

825

826 The name of this capability identifies its semantics of providing information on the operational
827 state of a managed Web service. This capability consists of two properties:

- 828 • `CurrentOperationalState` (mandatory)
- 829 • `LastOperationStateTransition` (zero to many)

830 and defines one event: `OperationalStateCapability`,

831 but applications may associate further properties and events with this capability.

832

833 The operational state model of a Web service endpoint used in this specification is the Web
834 service lifecycle (WSLC) state model as defined by the W3C Web Services Architecture
835 Management Task Force [**WSLC**]. Definition of the operational state in this specification uses the
836 transition paths for the service itself defined by the WSLC.

837

838 **5.2.5.1 Information markup declarations**

839 Each state MUST be identified by a QName and represented by a corresponding XML element.

840 Following is a list of elements corresponding to the operational states of the Web service

841 endpoint operation according to the WSLC state model [**WSLC**].

- 842 ▪ **UpState**

843 This element corresponds to the WSLC UP top-level state which means that the Web
844 service endpoint operation is capable of accepting new requests. This state is the parent
845 state of the BUSY and IDLE substates, as defined below.

- 846 ▪ **DownState**

847 This element corresponds to the WSLC DOWN top-level state which means that the Web
848 service endpoint operation is not capable of accepting new requests. This state is the
849 parent state of the STOPPED, CRASHED, and SATURATED substates, as defined
850 below.

- 851 ▪ **BusyState**
- 852 This element corresponds to the WSLC BUSY substate of UP which means that the Web
- 853 service endpoint operation is capable of accepting new requests during processing of
- 854 other requests. This element MUST contain the UpState element.
- 855 ▪ **IdleState**
- 856 This element corresponds to the WSLC IDLE substate of UP which means that the Web
- 857 service endpoint operation is capable of accepting new requests and is not processing
- 858 any other requests. This element MUST contain the UpState element.
- 859 ▪ **StoppedState**
- 860 This element corresponds to the WSLC STOPPED substate of DOWN which means that
- 861 the Web service endpoint operation is not capable of accepting new requests and was
- 862 intentionally stopped by an administrator. This element MUST contain the DownState
- 863 element.
- 864 ▪ **CrashedState**
- 865 This element corresponds to the WSLC CRASHED substate of DOWN which means that
- 866 the Web service endpoint operation is not capable of accepting new requests as a result
- 867 of some internal failure. This element MUST contain the DownState element
- 868 ▪ **SaturatedState**
- 869 This element corresponds to the WSLC SATURATED substate of DOWN which means
- 870 that the Web service endpoint operation is not capable of accepting new requests due to
- 871 lack of resources. This element MUST contain the DownState element.

872

873 It is possible to extend the above state model. Substates MAY be introduced and MUST be

874 identified by QNames, however, new top-level operational states MUST NOT be defined. In order

875 to represent the taxonomy lineage of substates in XML, the MUWS approach is used (section 3.2

876 in the **[MUWS]** part 2).

877

878 The **OperationalStateType** XML Schema type is declared as follows.

879

```
880 <xs:complexType name="OperationalStateType">
881 <xs:complexContent>
882     <xs:extension base="muws2:StateType"/>
883 </xs:complexContent>
884 </xs:complexType>
```

885

886 The **OperationalStateType** is used to declare elements which contain any valid elements

887 designating an operational state of a Web service endpoint.

888

- 889 ▪ A substate of the operational state MUST be declared according to the following rules.
- 890 ○ An XML element is declared with a QName which identifies the desired substate
- 891 semantics, for example my-app:DatabaseCleanupState
- 892 ○ The contents of the XML element MUST be the only element which corresponds
- 893 to the generalized state, for example mows:StoppedState

894

895 An instance of the request processing state information represented in XML may look as shown

896 in the following example,

897

```
898 <my:OperationalStateInformationElement xsi:type="mows:OperationalStateType">
899     <my-app:DatabaseCleanupState>
900         <mows:StoppedState>
901             <mows:DownState/>
902         </mows:StoppedState>
903     </my-app:DatabaseCleanupState>
904 </my:RequestProcessingStateInformationElement>
```

905

906 5.2.5.2 Properties

907 The following is the specification of the Web service endpoint operational state properties (i.e. the
908 XML elements which represent the state properties).

909

```
910 <CurrentOperationalState>mows:OperationalStateType</CurrentOperationalState>
911 <LastOperationalStateTransition>
912     muws2:StateTransitionType
913 </LastOperationalStateTransition> ?
```

914

915 **CurrentOperationalState** is the current operational state of the Web service endpoint being
916 managed. Metadata about this property is as follows.

- 917 ▪ Is *Mutable*
- 918 ▪ Is not *Modifiable*

919 **LastOperationalStateTransition** contains information about last operational state transition
920 which occurred at the Web service endpoint being managed. Metadata about this property is as
921 follows.

- 922 ▪ Is *Mutable*
- 923 ▪ Is not *Modifiable*

924

925 5.2.5.3 Events

926 The following specification defines this capability notification topics in the namespace mapped to
927 the **mowse** prefix.

928

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

930

931 **mowse:OperationalStateCapability** is a topic on which management events related to this
932 manageability capability SHOULD be emitted.

933

934 For information about changes of the operational state, a consumer MUST subscribe to
935 notifications on the changes of the CurrentOperationalState property (assuming that the
936 manageability endpoint implementation supports notifications about changes of this property).
937 Refer to **[WS-RP]** for information on how to subscribe to the property change notifications.

938

939 5.2.6 Operational Status

940 This capability is identified by the following URI:

941 **http://docs.oasis-open.org/mows-2/capabilities/OperationalStatus**

942 All properties, operations and events defined for this capability have the following metadata:

- 943 ▪ `<muws2:Capability>http://docs.oasis-open.org/mows-`
944 `2/capabilities/OperationalStatus</muws2:Capability>`

945

946 The name of this capability indicates its semantics of indicating the operational status of a
947 managed Web Service. This capability consists of one property:

- 948 • **OperationalStatus** (mandatory), as defined in the MUWS part 2 **OperationalStatus** capability
949 and defines one event: **OperationalStatusCapability**,
950 but applications may associate further properties and events with this capability.

951

952 WSDM manageable endpoints that intend to support the MUWS **Operational Status**
953 manageability capability (section 3.3 in the **[MUWS]** part 2) **MUST** abide by the following mapping
954 rules. When this capability support is indicated for a manageable endpoint, the mappings are in
955 effect.

956

957 The Web service lifecycle (WSLC) states defined by the W3C Web Services Architecture
958 Management Task Force **[WSLC]** map to the MUWS status values as follows:

- 959 ▪ The WSLC **UP** state **MUST** be reported as the **Available** contents of the
960 **muws2:OperationalStatus** property. Any sub-state of WSLC **UP** **MUST** be reported as
961 **Available**.
- 962 ▪ The WSLC **DOWN** state **MUST** be reported as the **Unavailable** contents of the
963 **muws2:OperationalStatus** property. Any sub-state of WSLC **DOWN** **SHOULD** be
964 reported as **Unavailable**. The **STOPPED** and **CRASHED** substates of WSLC **DOWN**
965 **MUST** be reported as **Unavailable**.
- 966 ▪ The WSLC **SATURATED** sub-state of **DOWN** **MAY** be reported as the
967 **PartiallyAvailable** contents of the **muws2:OperationalStatus** property.

968

969 5.2.6.1 Events

970 The following specification defines this capability notification topics in the namespace mapped to
971 the **mwse** prefix.

972

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

975

976 **mwse:OperationalStatusCapability** is a topic on which management events related to this
977 manageability capability **SHOULD** be emitted.

978

979 Property change events **MUST** be wrapped in Management events and published on topics
980 defined in this section.

981

982 5.2.7 Operation Operational Status

983 This capability is identified by the following URI:

984 **http://docs.oasis-open.org/wsdm/mows-2/capabilities/OperationOperationalStatus**

985 All properties, operations and events defined for this capability have the following metadata:

- 986 ▪ `<muws2:Capability>http://docs.oasis-open.org/wsdm/mows-`
987 `2/capabilities/OperationOperationalStatus</muws2:Capability>`

988 An operation operational status property reflects whether a named operation is available,
989 unavailable, or degraded. Operation operational status does not conform to a specific state
990 model. Operation operational status is related to the Request Processing Model of the [WSLC],
991 see Figure 6, but does not directly map to request processing states. The manageable resource
992 provides the appropriate mapping from request states to the status of an operation and sets the
993 *OperationOperationalStatus* property accordingly (see below).

994 The name of this capability identifies its semantics indicating the operational status of an
995 operation within a managed Web service. For each operation, there is one complex property that
996 provides the *OperationStatus* of that operation:

- 997 • *OperationOperationalStatus* (many)
998 • *OperationStatus*, as defined in section 5.2.6

999 and the capability defines one event: *OperationOperationalStatusCapability*

1000 but applications may associate further properties and events with this capability.

1001

1002 5.2.7.1 Properties

1003 The following is the specification of the Web service endpoint operation operational status
1004 complex property (i.e. XML element that represents the operational status of an operation).

```
1005 <OperationOperationalStatus operationName="xs:NCName", portType="xs:QName"? ...>*  
1006   <muws2:OperationalStatus>  
1007     (Available|PartiallyAvailable|Unavailable|Unknown)  
1008   </muws2:OperationalStatus>  
1009   {any} *  
1010 </OperationOperationalStatus>
```

1011 **OperationOperationalStatus** is the complex property representing each operation for which
1012 operational status information can be provided.

1013 **OperationOperationalStatus/@operationName** is the name of operation to which the
1014 operational status applies. This is a required attribute. The value of this attribute is the name of
1015 the operation as it is specified in the portType of the Web service

1016 **OperationOperationalStatus/@portType** is the QName of the portType in which the operation
1017 occurs. This is an optional attribute. The value of this attribute is the name of the portType as it is
1018 specified in the WSDL of the Web service.

1019 **OperationOperationalStatus/muws2:OperationalStatus** is the operational status property that
1020 is defined in MUWS Part 2 and used as well in the MOWS *OperationStatus* capability; see
1021 section 5.2.6. The valid values of this property with their intended interpretation in this context
1022 are:

- 1023 • *Available*: This value indicates that the named operation is operating normally within any
1024 configured operating parameters, and is able to perform its function. It is capable or
1025 receiving requests and completing processing.
1026 • *PartiallyAvailable*: This value indicates that the named operation is operating, but outside
1027 of configured operating parameters. An operation reporting this operation operational
1028 status is able to complete some requests but may fail others. For example, an operation

1029 may be failing only for a subset of requests because of database table required by only
1030 that subset is locked.
1031 • *Unavailable*: This value indicates that the named operation is not operating, and is not
1032 able to perform any functional tasks. The operation may be unable to received requests
1033 and it may be failing all requests.
1034 • *Unknown*: This value indicates that the named operation is unable to report status at this
1035 time.
1036

1037 5.2.7.2 Events

1038 The following specification defines this capability notification topics in the namespace mapped to
1039 the **mowse** prefix.

1040

```
1041 <wstop:Topic name="OperationOperationalStatusCapability"  
1042 messageTypes="muws1:ManagementEvent"/>
```

1043

1044 **mowse:OperationOperationalStatusCapability** is a topic on which management events related
1045 to this manageability capability SHOULD be emitted.

1046

1047 Property change events MUST be wrapped in Management events and published on topics
1048 defined in this section.

1049

1050 5.2.8 Request Processing State

1051 This capability is identified by the following URI:

1052 **<http://docs.oasis-open.org/mows-2/capabilities/RequestProcessingState>**

1053 All properties, operations and events defined for this capability have the following metadata:

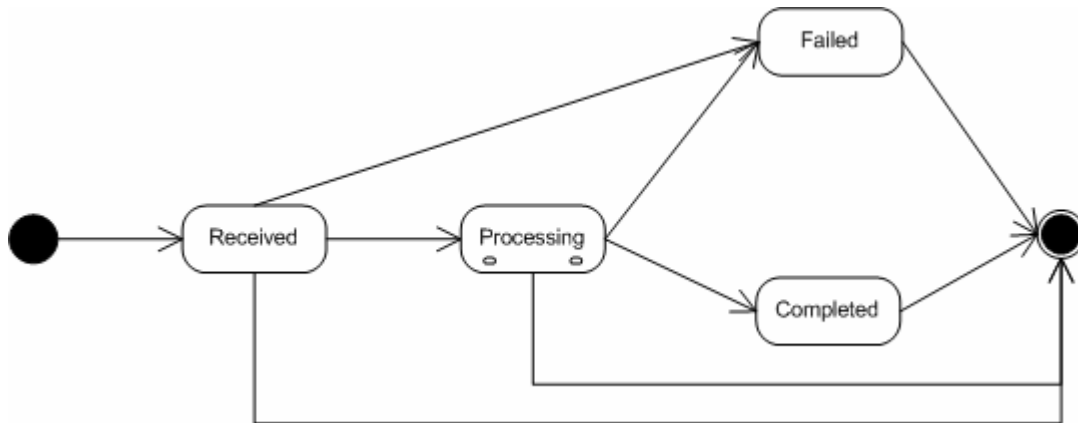
1054 • <muws2:Capability>[http://docs.oasis-open.org/mows-](http://docs.oasis-open.org/mows-2/capabilities/RequestProcessingState)
1055 [2/capabilities/RequestProcessingState](http://docs.oasis-open.org/mows-2/capabilities/RequestProcessingState)</muws2:Capability>

1056

1057 The name of this capability identifies its semantics of providing information on the processing
1058 state of a request to a managed Web service. This capability consists of no properties but defines
1059 12 events:

- 1060 • RequestProcessingObservations
- 1061 • RequestProcessingObservations/RequestReceived
- 1062 • RequestProcessingObservations/RequestProcessing
- 1063 • RequestProcessingObservations/RequestComplete
- 1064 • RequestProcessingObservations/RequestFailed
- 1065 • RequestProcessingObservations/Digest
- 1066 • RequestProcessingObservationsWithAttachments
- 1067 • RequestProcessingObservationsWithAttachments/RequestReceived
- 1068 • RequestProcessingObservationsWithAttachments/RequestProcessing
- 1069 • RequestProcessingObservationsWithAttachments/RequestComplete
- 1070 • RequestProcessingObservationsWithAttachments/RequestFailed

1071 • RequestProcessingObservationsWithAttachment/Digest
 1072 but applications may associate further properties and events with this capability.
 1073 By the definition, a Web service endpoint accepts and processes messages targeted at it –
 1074 *requests*. Every request goes through a number of states (e.g. received, processing, completed
 1075 or failed) as defined by the [WSLC] and extended here.
 1076



1077
 1078 **Figure 6.** Request processing states
 1079

1080 The state diagram represents a model in which states MAY have duration and transitions are
 1081 instantaneous. When extending this model one MUST extend only the Processing compound
 1082 state.
 1083

1084 5.2.8.1 Information markup declarations

1085 Each state MUST be identified by a QName and represented by a corresponding XML element.
 1086 Following is a list of elements corresponding to the top-level states of the request processing
 1087 state model (Figure 6).

- 1088 ▪ **RequestReceivedState**

1089 This element corresponds to the Received top-level state which means that the Web
 1090 service endpoint has accepted a request to perform one of the service's functional
 1091 responsibilities. This state represents the earliest point at which the manageability
 1092 provider knows that the request was dispatched to the Web service endpoint being
 1093 managed.

- 1094 ▪ **RequestProcessingState**

1095 This element corresponds to the Processing top-level state which means that the Web
 1096 service endpoint is doing some internal processing/execution to fulfill the requested
 1097 function. This state represents the earliest point at which the application module or
 1098 business logic begins processing the request. For example, if the application server
 1099 queues the request before dispatching it to the business logic, the time difference
 1100 between “request received” and “processing” will include the duration the request was
 1101 queued.

- 1102 ▪ **RequestCompletedState**

1103 This element corresponds to the Completed top-level state which means that the Web
 1104 service endpoint successfully completed requested function returning results to the
 1105 requester.

1106

- **RequestFailedState**

1107

This element corresponds to the Failed top-level state which means that the Web service endpoint encountered an error and didn't complete the requested function, returning error/fault to the requester.

1108

1109

1110

1111

It is possible to extend the above state model. Substates of the Processing top-level state MAY be introduced and MUST be identified by QNames, however, new top-level request processing states MUST NOT be defined. In order to represent the taxonomy lineage of substates in XML, the MUWS approach is used (section 3.2 in the [MUWS] part 2).

1112

1113

1114

1115

1116

The **RequestProcessingStateType** XML Schema type is declared as follows.

1117

1118

```
<xs:complexType name="RequestProcessingStateType">
```

1119

```
<xs:complexContent>
```

1120

```
  <xs:extension base="muws2:StateType"/>
```

1121

```
</xs:complexContent>
```

1122

```
</xs:complexType>
```

1123

1124

The **RequestProcessingStateType** is used to declare elements which designate a request processing state – top-level or substates of the Processing.

1125

1126

1127

A substate of the Processing compound state MUST be declared according to the following rules.

1128

An XML element is declared with a QName which identifies the desired substate semantics, for example my-soap:SerializationState

1129

1130

The contents of the XML element MUST be the only element which corresponds to the generalized state, for example muws2:RequestProcessingState

1131

1132

1133

An instance of the request processing state information represented in XML may appear as shown in the following example,

1134

1135

1136

```
<my:RequestProcessingStateInformationElement
```

1137

```
  xsi:type="mows:RequestProcessingStateType">
```

1138

```
    <my-soap:SerializationState>
```

1139

```
      <mows:RequestProcessingState/>
```

1140

```
    </my-soap:SerializationState>
```

1141

```
</my:RequestProcessingStateInformationElement>
```

1142

1143

5.2.8.2 Events

1144

Notifications are emitted when requests enter one of the request processing states (Figure 6).

1145

1146

Property change events MUST be wrapped in Management events and published on topics defined in this section.

1147

1148

1149

The following specification defines the Web service endpoint request processing state notification topics in the namespace mapped to the **mowse** prefix. The message patterns' expression and dialect MUST match precisely what is declared below.

1150

1151

1152

```
1153 <wstop:Topic name="RequestProcessingStateCapability"  
1154 messageTypes="muws1:ManagementEvent"/>  
1155  
1156 <wstop:Topic name="RequestProcessingObservations"  
1157   messageTypes="muws1:ManagementEvent">  
1158   <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1159   //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1160   and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1161   </wstop:MessagePattern>  
1162   <wstop:Topic name="RequestReceived"  
1163     messageTypes="muws1:ManagementEvent">  
1164     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1165     //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1166     and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1167     </wstop:MessagePattern>  
1168     </wstop:Topic>  
1169     <wstop:Topic name="RequestProcessing"  
1170       messageTypes="muws1:ManagementEvent">  
1171       <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1172       //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1173       and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1174       </wstop:MessagePattern>  
1175       </wstop:Topic>  
1176       <wstop:Topic name="RequestCompleted"  
1177         messageTypes="muws1:ManagementEvent">  
1178         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1179         //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1180         and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1181         </wstop:MessagePattern>  
1182         </wstop:Topic>  
1183         <wstop:Topic name="RequestFailed"  
1184           messageTypes="muws1:ManagementEvent">  
1185           <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1186           //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1187           and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1188           </wstop:MessagePattern>  
1189           </wstop:Topic>  
1190           <wstop:Topic name="Digest"  
1191             messageTypes="muws1:ManagementEvent">  
1192             <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1193             //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1194             and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1195             </wstop:MessagePattern>  
1196             </wstop:Topic>  
1197           </wstop:Topic>  
1198         </wstop:Topic>  
1199     <wstop:Topic name="RequestProcessingObservationsWithAttachments"  
1200       messageTypes="muws1:ManagementEvent">  
1201       <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">  
1202       //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation  
1203       and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]  
1204       </wstop:MessagePattern>  
1205       <wstop:Topic name="RequestReceived"  
1206         messageTypes="muws1:ManagementEvent">
```

```

1207     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
1208 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
1209 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
1210     </wstop:MessagePattern>
1211     </wstop:Topic>
1212     <wstop:Topic name="RequestProcessing"
1213         messageTypes="muws1:ManagementEvent">
1214     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
1215 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
1216 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
1217     </wstop:MessagePattern>
1218     </wstop:Topic>
1219     <wstop:Topic name="RequestCompleted"
1220         messageTypes="muws1:ManagementEvent">
1221     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
1222 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
1223 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
1224     </wstop:MessagePattern>
1225     </wstop:Topic>
1226     <wstop:Topic name="RequestFailed"
1227         messageTypes="muws1:ManagementEvent">
1228     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
1229 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
1230 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
1231     </wstop:MessagePattern>
1232     </wstop:Topic>
1233     <wstop:Topic name="Digest"
1234         messageTypes="muws1:ManagementEvent">
1235     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
1236 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
1237 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
1238     </wstop:MessagePattern>
1239     </wstop:Topic>
1240 </wstop:Topic>

```

1241

1242 **mowse:RequestProcessingStateCapability** is a topic on which management events related to
1243 this manageability capability SHOULD be emitted.

1244 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations** indicates availability
1245 of any information about the processing of any request by the Web service endpoint (Figure 6) as
1246 observed by the implementation of a manageable Web service.

1247 The notification message for this topic MUST contain at most one
1248 **RequestProcessingNotification** element (defined in section 5.2.8.2.1). The MUWS
1249 management event MUST also declare the event situation category with the
1250 muws2:ReportSituation element and the severity value "1" (Informational). It is
1251 recommended to subscribe to these topics with proper preconditions and selectors using
1252 expressions against the contents of the RequestProcessingNotification element.

1253 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:RequestRe**
1254 **ceived** indicates that a request was received by the Web service endpoint being managed
1255 (Received state in Figure 6). The notification message format for this topic is the same as the
1256 notification message format for the
1257 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic. This is a state
1258 change event and therefore notification messages MUST contain exactly one

1259 muws2:StateTransition element inside of the RequestProcessingNotification/StateInformation
1260 element.

1261 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:RequestPro**
1262 **cessing** indicates that a request is being processed by the Web service endpoint being managed
1263 (Processing state in Figure 6). The notification message format for this topic is the same as the
1264 notification message format for the
1265 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic. This is a state
1266 change event and therefore notification messages MUST contain exactly one
1267 muws2:StateTransition element inside of the RequestProcessingNotification/StateInformation
1268 element.

1269 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:RequestCo**
1270 **mpleted** indicates that a request was successfully completed by the Web service endpoint being
1271 managed (Completed state in Figure 6). The notification message format for this topic is the
1272 same as the notification message format for the
1273 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic. This is a state
1274 change event and therefore notification messages MUST contain exactly one
1275 muws2:StateTransition element inside of the RequestProcessingNotification/StateInformation
1276 element.

1277 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:RequestFail**
1278 **ed** indicates that a request was failed (not successfully completed) by the Web service endpoint
1279 being managed (Failed state in Figure 6). The notification message format for this topic is the
1280 same as the notification message format for the
1281 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic. This is a state
1282 change event and therefore notification messages MUST contain exactly one
1283 muws2:StateTransition element inside of the RequestProcessingNotification/StateInformation
1284 element.

1285 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:Digest**
1286 indicates availability of summary information about a request processed by the Web service
1287 endpoint being managed. The notification message format for this topic is the same as the
1288 notification message format for the
1289 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic. This is a digest
1290 event and therefore notification messages MUST contain one or more muws2:StateTransition
1291 elements inside of the RequestProcessingNotification/StateInformation element. Each
1292 muws2:StateTransition element describes a state transition which occurred with that one request
1293 which this summary notification is informing about. Each state transition information element
1294 carries an attribute indicating the time when that particular transition occurred. Using this
1295 information the manageability consumer can reconstruct the sequence of events with regards to
1296 the request.

1297 **mowse:ManageableEndpoint/mowse:RequestProcessingObservationsWithAttachments**
1298 topic and all of its subtopics are defined exactly as the
1299 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic and its respective
1300 subtopics, except that the notification messages MUST include attachments (if any) of the
1301 request and reply messages sent to/from the Web service endpoint being managed.

1302 The notification message format for this topic and all of its subtopics is the same as the
1303 notification message format for the
1304 mowse:ManageableEndpoint/mowse:RequestProcessingObservations topic, except that
1305 attachments may be sent along with the message. The precise mechanism of sending
1306 the attachment is dependent on 1) the binding of the notification consumer Web service
1307 endpoint **[WS-N]** and 2) the binding of the Web service endpoint being managed.

1308

1309 The mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:
1310 RequestProcessing topic MAY be extended with custom subtopics in order to represent custom
1311 request processing substates of the Processing compound state (Figure 6).

1312

1313 Note that the result of the message pattern XPath expressions in the topic declarations above is
1314 the XML nodeset **[XPath]** of the notification messages that are sent inside of the S:Body element
1315 or the wsnt:Notify element **[WS-N]**.

1316

1317 Note that for the XPath expressions defined here the prefix-to-namespace mapping context
1318 MUST include all prefixes which appear in the XPath expression and mapped according to the
1319 table in the section 5.

1320

1321 **5.2.8.2.1 RequestProcessingNotification message**

1322 The RequestProcessingNotification message format is defined as follows.

1323

```
1324 <RequestProcessingNotification CurrentTime="xs:dateTime" ...>  
1325 <Request ...>  
1326   <TransportInformation ...> {any}* </TransportInformation> ?  
1327   <Message ...>  
1328     <Size Unit=("bit" | "byte" | "word" | "dword" | "qword")  
1329       ...>xs:positiveInteger</Size> ?  
1330     (  
1331       <NotIncluded/> |  
1332       <Text>xs:string</Text> |  
1333       <Binary>xs:base64Binary</Binary> |  
1334       <Xml>{any}* </Xml>  
1335     )  
1336     {any}*  
1337   </Message>  
1338   {any}*  
1339 </Request> ?  
1340 <Reply ...>  
1341   <!-- ... see contents of the Request element above ... -->  
1342 </Reply> ?  
1343 <StateInformation>  
1344 <muws2:StateTransition> <!-- ...see [MUWS]... --> </muws2:StateTransition> +  
1345 </StateInformation>  
1346 {any}*  
1347 </RequestProcessingNotification>
```

1348

1349 **RequestProcessingNotification** is a container element of the information about a request going
1350 through the request processing states (Figure 6).

1351 **RequestProcessingNotification/@CurrentTime** indicates current time measured at the
1352 manageability endpoint. All time/date values in this notification information are synchronized with
1353 this time indication.

1354 **RequestProcessingNotification/Request** element contains information about the request itself.
1355 Note that the request is not necessarily serialized as a SOAP message. Therefore, the contents
1356 allow information about requests in general, however the information has to be serializable in
1357 XML **[XML]**. The presence of this element in the notification MUST indicate presence of the

1358 actual request message sent to the Web service endpoint being managed. The contents may
1359 vary depending on what the implementation of the manageability endpoint can or intends to
1360 provide. For example, for security reasons the actual contents of the message may be omitted.
1361 However, in order to indicate that the request message exists, this element has to be included in
1362 the notification.

1363 **RequestProcessingNotification/Request/TransportInformation** element contains information
1364 about the transport by which the request was received. The content of this element is open, but
1365 WSDM defines the following elements useable for TCP/IP transports.

```
1366 <TcpIpInfo  
1367     Direction=("from" | "to")  
1368     Port="xs:positiveInteger"  
1369     Protocol=("TCP" | "UDP") ...>  
1370     (  
1371     <IPV4Address>  
1372         xs:hexBinary[xs:length[@value="8" and @fixed="true"]]  
1373     </IPV4Address> |  
1374     <IPV6Address>  
1375         xs:hexBinary[xs:length[@value="32" and @fixed="true"]]  
1376     </IPV6Address>  
1377     )  
1378     {any}*  
1379 </TcpIpInfo>
```

1380 **TcpIpInfo** contains information about a communication to or from an IP addressable
1381 network device.

1382 **TcpIpInfo/@Direction** indicates communication to or from the IP addressable network
1383 device.

1384 **TcpIpInfo/@Port** is a TCP/IP network port number used on the IP addressable network
1385 device.

1386 **TcpIpInfo/@Protocol** indicates if the TCP or UDP protocol is used.

1387 **TcpIpInfo/IPV4Address** contains hexadecimal representation of the IP address version
1388 4. The value MUST represent 32 bits.

1389 **TcpIpInfo/IPV6Address** contains hexadecimal representation of the IP address version
1390 6. The value MUST represent 128 bits.

1391 **RequestProcessingNotification/Request/Message** element contains the message observed by
1392 the Web service endpoint being managed.

1393 **RequestProcessingNotification/Request/Message/Size** indicates size of the message. When
1394 subscribed to observations with attachments, this value includes the size of the message payload
1395 plus all the attachments. Otherwise, just the payload of the message (i.e. size of the contents of
1396 the RequestProcessingNotification/Request/Message element) is reported. Note that the actual
1397 message contents may not be reported for security reasons, however size may be reported.

1398 **RequestProcessingNotification/Request/Message/Size/@Unit** indicates what units were used
1399 to calculate the size of the message. The valid values of this attribute are:

1400 **bit** – size indicates number of bits in the message.

1401 **byte** – size indicates number of bytes (8 bit sets) in the message

1402 **word** – size indicates number of double bytes (16 bit sets) in the message.

1403 **dword** – size indicates number of double words (32 bit sets) in the message.

1404 **qword** – size indicates number of quad words (64 bit sets) in the message.

1405 **RequestProcessingNotification/Request/Message/NotIncluded** element indicates that the
1406 message content is intentionally not provided by the implementation of the Web service endpoint
1407 manageability.

1408 **RequestProcessingNotification/Request/Message/Text** element contains the observed
1409 message's text representation. For example, a non-well formed XML message should be
1410 represented as text. It is recommended that text data is wrapped in an XML CDATA section
1411 [XML].

1412 **RequestProcessingNotification/Request/Message/Binary** element contains the binary
1413 representation of the observed message. If a message cannot be represented as either well-
1414 formed XML nor as text, it should be binary encoded.

1415 **RequestProcessingNotification/Request/Message/XML** element contains the observed
1416 message's XML representation. For example, a SOAP message envelope element (S:Envelope)
1417 may appear in the contents.

1418 **RequestProcessingNotification/Request/{any}** is an extensibility element where additional
1419 information about the request MAY appear. The form of the information representation in XML is
1420 manageability endpoint implementation specific. In other words, vendor extensions may appear
1421 here.

1422 The **RequestProcessingNotification/Reply** element contains information about the reply (if any)
1423 for the request. Note that fault is also a valid reply element. The content of this element has the
1424 same format as the RequestProcessingNotification/Request element.

1425 **RequestProcessingNotification/StateInformation** element contains information about the
1426 request processing state.

1427 **RequestProcessingNotification/StateInformation/muws2:StateTransition** element contains
1428 information about a state transition. There MUST be exactly one such element for each state
1429 change event. There MUST be one or more such elements for the digest event.

1430 **RequestProcessingNotification/StateInformation/muws2:StateTransition/@muws2:Time**
1431 indicates time when the described transition occurred. Note that according to the request
1432 processing state model (Figure 6), all transitions are instantaneous. Time is measured at the
1433 implementation of the manageability endpoint and is synchronized with the
1434 RequestProcessingNotification/@CurrentTime value reading.

1435 **RequestProcessingNotification/StateInformation/muws2:StateTransition/muws2:EnteredSt**
1436 **ate** indicates which request processing state was entered.

1437 **RequestProcessingNotification/StateInformation/muws2:StateTransition/muws2:Previous**
1438 **State** indicates which request processing state was exited.

1439 **RequestProcessingNotification/{any}** is an extensibility element where additional information
1440 about this request processing notification MAY appear. The form of the information representation
1441 in XML is manageability endpoint implementation specific. In other words, vendor extensions may
1442 appear here.

1443 The contents of the RequestProcessingNotification element SHOULD be used to specify
1444 selectors [WS-N] when subscribing to notification messages containing this element.

1445 **5.2.8.2.2 Examples of events against the Web service endpoint request** 1446 **processing state**

1447 Consider the following message exchange with a fictitious order-entry Web service endpoint.

1448
1449 Request:

```
1450 <S:Envelope xmlns:x="..." ... >
1451 . . .
1452 <S:Body>
```

```
1453     <x:Order>
1454         <x:Item>...</x:Item>
1455         <x:Quantity>...</x:Quantity>
1456     </x:Order>
1457 </S:Body>
1458 </S:Envelope>
```

1459
1460 Reply:

```
1461 <S:Envelope xmlns:x="..." ... >
1462 . . .
1463 <S:Body>
1464     <x:Shipped>
1465         <x:Item>...</x:Item>
1466         <x:Quantity>...</x:Quantity>
1467     </x:Shipped>
1468 </S:Body>
1469 </S:Envelope>
```

1470
1471 To be notified of a particular item shortage when the order request is processed and the shipped
1472 quantity is less than the ordered quantity, the following XPath selector should be specified when
1473 subscribing to the
1474 **mowse:ManageableEndpoint/mowse:RequestProcessingObservations/mowse:RequestCo**
1475 **mpleted** topic.

1476
1477 Selector:
1478 `boolean(//mows:RequestProcessingNotification[mows:Request/mows:Message/mows:Xml/x:Ord`
1479 `er/x:Quantity < mows:Reply/mows:Message/mows:Xml/x:Shipped/x:Quantity)`

1480
1481 This way, when the condition is met, the manageable Web service endpoint will emit the
1482 notification message containing the **RequestProcessingNotification** element with the following
1483 contents.

```
1484  
1485 <RequestProcessingNotification CurrentTime="...">
1486 <Request>
1487     <TransportInformation>
1488         <TcpIpInfo Direction="from" Port="2840" Protocol="TCP">
1489             <IPV4Address>C0A80002</IPV4Address>
1490         </TcpIpInfo>
1491         <TcpIpInfo Direction="to" Port="80" Protocol="TCP">
1492             <IPV4Address>C0A80003</IPV4Address>
1493         </TcpIpInfo>
1494     </TransportInformation>
1495     <Message>
1496         <Size Unit="byte">257</Size>
1497         <Xml>
1498             <S:Envelope xmlns:S="..." xmlns:x="..." ...>
1499             . . .
1500             <S:Body>
1501                 <x:Order>
1502                     <x:Item>123</x:Item>
1503                     <x:Quantity>10</x:Quantity>
```

```

1504         </x:Order>
1505     </S:Body>
1506 </S:Envelope>
1507 </Xml>
1508 </Message>
1509 </Request>
1510 <Reply>
1511     <TransportInformation>
1512         <TcpIpInfo Direction="to" Port="2840" Protocol="TCP">
1513             <IPv4Address>C0A80002</IPv4Address>
1514         </TcpIpInfo>
1515         <TcpIpInfo Direction="from" Port="80" Protocol="TCP">
1516             <IPv4Address>C0A80003</IPv4Address>
1517         </TcpIpInfo>
1518     </TransportInformation>
1519     <Message>
1520     <Size Unit="byte">232</Size>
1521     <Xml>
1522     <S:Envelope xmlns:S="..." xmlns:x="..." ...>
1523     . . .
1524     <S:Body>
1525         <x:Shipped>
1526             <x:Item>123</x:Item>
1527             <x:Quantity>2</x:Quantity>
1528         </x:Shipped>
1529     </S:Body>
1530 </S:Envelope>
1531 </Xml>
1532 </Message>
1533 </Reply>
1534 <muws2:StateTransition Time="...">
1535 <muws2:EnteredState/><RequestCompletedState/></muws2:EnteredState>
1536 <muws2:PreviousState><RequestProcessingState/></muws2:PreviousState>
1537 </muws2:StateTransition>
1538 . . .
1539 </RequestProcessingNotification>
1540
1541

```

1542 6 References

1543 6.1 Normative

- 1544 [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
1545 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 1546 [MUWS] V. Bullard, et al., *Web Services Distributed Management:Management
1547 using Web Services (MUWS 1.1) Part 1*, [http://docs.oasis-
1549 open.org/wsdm/wsdm-muws1-1.1-os-01.pdf](http://docs.oasis-
1548 open.org/wsdm/wsdm-muws1-1.1-os-01.pdf)
- 1550 V. Bullard, et al., *Web Services Distributed Management:Management
1551 using Web Services (MUWS 1.1) Part 2*, [http://docs.oasis-
1553 open.org/wsdm/wsdm-muws2-1.1-os-01.pdf](http://docs.oasis-
1552 open.org/wsdm/wsdm-muws2-1.1-os-01.pdf)
- 1554 [WS-A] Don Box, et al., *Web services Addressing (WS-Addressing)*,
<http://www.w3.org/TR/ws-addr-core>
- 1555 [WS-RP] Steve Graham, et al., *Web Services Resource Properties 1.2 (WS-
1556 ResourceProperties)*, [http://docs.oasis-open.org/wsr/wsr-
1558 ws_resource_properties-1.2-spec-os-01.pdf](http://docs.oasis-open.org/wsr/wsr-
1557 ws_resource_properties-1.2-spec-os-01.pdf)
- 1559 [WS-N] Steve Graham, et al., *Web Services Base Notification 1.2 (WS-
1560 BaseNotification)*, [http://docs.oasis-open.org/wsn/wsn-
1562 ws_base_notification-1.3-spec-pr-03.pdf](http://docs.oasis-open.org/wsn/wsn-
1561 ws_base_notification-1.3-spec-pr-03.pdf)
- 1563 [WSDL] William Vambenepe, *Web Services Topics 1.2 (WS-Topics)*,
http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-pr-02.pdf
- 1564 [WSDL] Erik Christensen, et al., *Web services Description Language (WSDL)
1565 1.1*, W3C Note, March 2001, <http://www.w3.org/TR/wsd>
- 1566 [SOAP] Don Box, et al., *Simple Object Access Protocol 1.1*, W3C Note, May
2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- 1567 [XMLS] Henry S. Thompson, et al. *XML Schema Part 1: Structures*, W3C
1568 Recommendation, May 2001, <http://www.w3.org/TR/xmlschema-1/>
- 1569 Paul V. Biron, et al. *XML Schema Part 2: Datatypes*, W3C
1570 Recommendation, May 2001, <http://www.w3.org/TR/xmlschema-2/>
- 1571 [XML] Tim Bray, et al., *Extensible Markup Language (XML) 1.0 (Third Edition)*,
1572 W3C Recommendation, February 2004, <http://www.w3.org/TR/REC-xml>
- 1573 [XNS] Tim Bray, et al., *Namespaces in XML*, W3C Recommendation, January
1999, <http://www.w3.org/TR/REC-xml-names/>
- 1574 [XPath] James Clark, et al., *XML Path Language (XPath) Version 1.0*, W3C
1575 Recommendation, November 1999, [http://www.w3.org/TR/1999/REC-
xpath-19991116](http://www.w3.org/TR/1999/REC-
1576 xpath-19991116)

1577 6.2 Non-normative

- 1578 [MOWS-Reqs] Mark Potts, et al., *WSDM Management of Web Services Requirements*,
1579 October 2003, [http://www.oasis-
open.org/apps/org/workgroup/wsdm/download.php/3887/WSDM-MOWS-
Requirements.20031008.doc](http://www.oasis-
1580 open.org/apps/org/workgroup/wsdm/download.php/3887/WSDM-MOWS-
1581 Requirements.20031008.doc)
- 1582 [WS-Arch] David Booth, et al. *Web Services Architecture*, W3C Working Group
1583 Note, February 2004, [http://www.w3.org/TR/2004/NOTE-ws-arch-
20040211/](http://www.w3.org/TR/2004/NOTE-ws-arch-
1584 20040211/)
- 1585 [WSLC] Hao He, et al., *Web Service Management: Service Lifecycle*, W3C Note,
1586 February 2004, <http://www.w3.org/TR/2004/NOTE-wslc-20040211/>

1587 **[WSS]** Anthony Nadalin, et al., *Web Services Security: SOAP Message Security*
1588 *1.0 (WS-Security 2004)*, March 2004, OASIS Standard, [http://docs.oasis-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
1589 [open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
1590 **[BSP]** Abbie Barbir, et al., *Basic Security Profile Version 1.0*, WS-I Working
1591 Group Draft, May 2004, [http://www.ws-i.org/Profiles/BasicSecurityProfile-](http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2004-05-12.html)
1592 [1.0-2004-05-12.html](http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2004-05-12.html)
1593

1594

Appendix A. Acknowledgments

1595 The following people made contributions to the WSDM MOWS Version 1.1 specification: Brian
1596 Carroll, Fred Carter, John DeCarlo, Andreas Dharmawan, Maryann Hondo, Heather Kreger,
1597 Bryan Murray, Michael Perks, Igor Sedukhin, William Vambenepe, Andrea Westerinen.

1598 The following individuals were members of the committee when the specification 1.0 version was
1599 approved by the technical committee: Guru Bhat, Jeff Bohren, Winston Bumpus, Nick Butler,
1600 Brian Carroll, Fred Carter, Michael Clements, David Cox, John DeCarlo, Andreas Dharmawan,
1601 Mark Ellison, John Fuller, Paul Lipton, Heather Kreger, Hal Lockhart, Frederico Maciel, Tom
1602 Maguire, Bryan Murray, Richard Nikula, Mark Peel, Richard Pelavin, Homayoun Pourheidari,
1603 Warren Roberts, Karl Schopmeyer, Igor Sedukhin, David Snelling, Thomas Studwell, William
1604 Vambenepe, Andrea Westerinen, Jim Willits, Zhili Zhang.

1605

1606 The following people made contributions to the WSDM MOWS Version 1.1 specification: Barry
1607 Atkins, Vaughn Bullard, Fred Carter, David Cox, Mark Ellison, Heather Kreger, Fred Maciel,
1608 Bryan Murray, Kirk Wilson with special thanks to Kirk Wilson and Mark Ellison as editors.

1609 The following individuals were members of the committee while the specification 1.1 version was
1610 developed and approved by the technical committee: Guru Bhat, Jeff Bohren, Vaughn Bullard,
1611 Winston Bumpus, Fred Carter, Michael Clements, David Cox, Zulah Eckert, Mark Ellison, John
1612 Fuller, Tony Gullotta, Heather Kreger, Richard Landau, Frederico Maciel, Tom Maguire, David
1613 Melgar, Bryan Murray, Richard Nikula, Mark Peel, Mitsunori Satomi, Thomas Studwell, William
1614 Vambenepe, Kirk Wilson, Zhili Zhang.

Appendix B. Revision History

Rev	Date	By Whom	
wd	2003-10-31	Igor Sedukhin	
wd	2003-11-14	Igor Sedukhin	
wd	2003-12-02	Igor Sedukhin	
wd	2004-01-26	Igor Sedukhin	
wd	2004-02-17	Igor Sedukhin	
wd	2004-03-01	Igor Sedukhin	
wd	2004-03-18	Igor Sedukhin	
wd	2004-03-19	Igor Sedukhin	
wd	2004-03-24	Igor Sedukhin	
wd	2004-03-24	Igor Sedukhin	
cd	2004-04-02	Igor Sedukhin	
wd	2004-07-21	Igor Sedukhin	
wd	2004-09-11	Igor Sedukhin	
wd	2004-10-11	Igor Sedukhin	
wd	2004-10-24	Igor Sedukhin	
wd	2004-11-04	Igor Sedukhin	
wd	2004-11-15	Igor Sedukhin	
wd	2004-11-19	Igor Sedukhin	
wd	2004-11-23	Igor Sedukhin	
wd	2004-12-03	Igor Sedukhin	
cd	2004-12-10	Igor Sedukhin	
standard	2005-03-09	Igor Sedukhin	
wd v 1.1	2005-10-27	Kirk Wilson	Addition of new capability: OperationMetric Creation of Appendix A
wd v 1.1 # 3	2005-11-03	Kirk Wilson	Addition of new Metric properties Version 1.1 namespace and file names added
wd v 1.1 #4	2005-11-04	Kirk Wilson	Addition of new capability: OperationOperationalStatus
wd v 1.1 #5	2005-11-16	Kirk Wilson	Restructure treatment of OperationMetrics and

Rev	Date	By Whom	
			OperationOperationalStatus Remove pseudo-UML diagrams Editorial corrections (result of F2F review)
wd v 1.1 #6	2005-11-28	Kirk Wilson	Additional editorial corrections Delete section 2.4 (UML "mind-map")
wd v 1.1 #7	2006-01-27	Kirk Wilson	Schema syntax check. Added acknowledgements
Final wd	2006-01-27	Kirk Wilson	Prepare final WD for TC vote
cd v 1.1 #1	2006-02-23	Kirk Wilson	Committee Draft candidate

1616

1617

Appendix C. Notices

1618 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1619 that might be claimed to pertain to the implementation or use of the technology described in this
1620 document or the extent to which any license under such rights might or might not be available;
1621 neither does it represent that it has made any effort to identify any such rights. Information on
1622 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1623 website. Copies of claims of rights made available for publication and any assurances of licenses
1624 to be made available, or the result of an attempt made to obtain a general license or permission
1625 for the use of such proprietary rights by implementors or users of this specification, can be
1626 obtained from the OASIS Executive Director.

1627 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1628 applications, or other proprietary rights which may cover technology that may be required to
1629 implement this specification. Please address the information to the OASIS Executive Director.

1630 Copyright © OASIS Open 2003-2006. *All Rights Reserved.*

1631 This document and translations of it may be copied and furnished to others, and derivative works
1632 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1633 published and distributed, in whole or in part, without restriction of any kind, provided that the
1634 above copyright notice and this paragraph are included on all such copies and derivative works.
1635 However, this document itself does not be modified in any way, such as by removing the
1636 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS
1637 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1638 Property Rights document must be followed, or as required to translate it into languages other
1639 than English.

1640 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1641 successors or assigns.

1642 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1643 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1644 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
1645 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1646 PARTICULAR PURPOSE.

1647

Appendix D. XML Schemas

```

1649 <?xml version="1.0" encoding="utf-8"?>
1650 <xs:schema targetNamespace="http://docs.oasis-open.org/wsdm/mows-2.xsd"
1651     xmlns:mows="http://docs.oasis-open.org/wsdm/mows-2.xsd"
1652     xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
1653     xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
1654     xmlns:wsa="http://www.w3.org/2005/08/addressing"
1655     xmlns:xs="http://www.w3.org/2001/XMLSchema"
1656     elementFormDefault="qualified" attributeFormDefault="unqualified">
1657
1658     <xs:import namespace="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
1659 schemaLocation="http://docs.oasis-open.org/wsdm/muws1-2.xsd"/>
1660     <xs:import namespace="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
1661 schemaLocation="http://docs.oasis-open.org/wsdm/muws2-2.xsd"/>
1662     <xs:import namespace="http://www.w3.org/2005/08/addressing"
1663 schemaLocation="http://www.w3.org/2005/08/addressing/ws-addr.xsd"/>
1664
1665     <!-- MOWS::ManageabilityReferences -->
1666     <xs:element name="GetManageabilityReferences"/>
1667     <xs:element name="GetManageabilityReferencesResponse">
1668         <xs:complexType>
1669             <xs:sequence>
1670                 <xs:element ref="muws1:ManageabilityEndpointReference"
1671                     maxOccurs="unbounded"/>
1672             </xs:sequence>
1673         </xs:complexType>
1674     </xs:element>
1675
1676     <!-- MOWS::Identification -->
1677     <xs:element name="EndpointReference" type="wsa:EndpointReferenceType"/>
1678     <xs:element name="EndpointDescriptions">
1679         <xs:complexType>
1680             <xs:sequence>
1681                 <xs:element name="description" type="xs:anyURI"
1682                     minOccurs="0" maxOccurs="unbounded"/>
1683             </xs:sequence>
1684             <xs:anyAttribute namespace="##other" processContents="lax"/>
1685         </xs:complexType>
1686     </xs:element>
1687
1688     <!-- Operation Properties Attribute Group -->
1689     <xs:attributeGroup name="OperationNameGroup">
1690         <xs:attribute name="operationName" type="xs:NCName" use="required"/>
1691         <xs:attribute name="portType" type="xs:QName" use="optional"/>
1692     </xs:attributeGroup>
1693
1694     <!-- MOWS Operation Operation Status Type -->
1695     <xs:complexType name="OperationOperationalStatusType">
1696         <xs:sequence>
1697             <xs:element ref="muws2:OperationalStatus" />
1698             <xs:any namespace="##any" processContents="lax"
1699                 minOccurs="0" maxOccurs="unbounded"/>

```

```

1700         </xs:sequence>
1701         <xs:attributeGroup ref="OperationNameGroup"/>
1702         <xs:anyAttribute namespace="##other" processContents="lax"/>
1703     </xs:complexType>
1704
1705     <xs:element name="OperationOperationalStatus"
1706               type="OperationOperationalStatusType" />
1707
1708     <!-- MOWS::Metrics -->
1709     <xs:complexType name="IntegerCounter">
1710         <xs:simpleContent>
1711             <xs:extension base="xs:nonNegativeInteger">
1712                 <xs:attributeGroup ref="muws2:MetricAttributes"/>
1713                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1714             </xs:extension>
1715         </xs:simpleContent>
1716     </xs:complexType>
1717
1718     <xs:complexType name="DurationMetric">
1719         <xs:simpleContent>
1720             <xs:extension base="xs:duration">
1721                 <xs:attributeGroup ref="muws2:MetricAttributes"/>
1722                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1723             </xs:extension>
1724         </xs:simpleContent>
1725     </xs:complexType>
1726
1727     <xs:complexType name="OperationIntegerCounter">
1728         <xs:simpleContent>
1729             <xs:extension base="mows:IntegerCounter">
1730                 <xs:attributeGroup ref="OperationNameGroup"/>
1731             </xs:extension>
1732         </xs:simpleContent>
1733     </xs:complexType>
1734
1735     <xs:complexType name="OperationDurationMetric">
1736         <xs:simpleContent>
1737             <xs:extension base="mows:DurationMetric">
1738                 <xs:attributeGroup ref="OperationNameGroup"/>
1739             </xs:extension>
1740         </xs:simpleContent>
1741     </xs:complexType>
1742
1743     <xs:element name="NumberOfRequests" type="mows:IntegerCounter"/>
1744     <xs:element name="NumberOfSuccessfulRequests" type="mows:IntegerCounter"/>
1745     <xs:element name="NumberOfFailedRequests" type="mows:IntegerCounter"/>
1746     <xs:element name="ServiceTime" type="mows:DurationMetric"/>
1747     <xs:element name="MaxResponseTime" type="mows:DurationMetric"/>
1748     <xs:element name="LastResponseTime" type="mows:DurationMetric"/>
1749     <xs:element name="MaxRequestSize" type="mows:IntegerCounter"/>
1750     <xs:element name="LastRequestSize" type="mows:IntegerCounter"/>
1751     <xs:element name="MaxReponseSize" type="mows:IntegerCounter"/>
1752     <xs:element name="LastReponseSize" type="mows:IntegerCounter"/>
1753
1754     <!-- MOWS::Operation Metrics -->

```



```

1755 <xs:complexType name="OperationMetricType">
1756   <xs:sequence>
1757     <xs:element ref="NumberOfRequests" minOccurs="0"/>
1758     <xs:element ref="NumberOfSuccessfulRequests"
1759       minOccurs="0"/>
1760     <xs:element ref="NumberOfFailedRequests" minOccurs="0"/>
1761     <xs:element ref="ServiceTime" minOccurs="0"/>
1762     <xs:element ref="MaxResponseTime" minOccurs="0"/>
1763     <xs:element ref="LastResponseTime" minOccurs="0"/>
1764     <xs:element ref="MaxRequestSize" minOccurs="0"/>
1765     <xs:element ref="LastRequestSize" minOccurs="0"/>
1766     <xs:element ref="MaxResponseSize" minOccurs="0"/>
1767     <xs:element ref="LastResponseSize" minOccurs="0"/>
1768     <xs:any namespace="##any" processContents="lax"
1769       minOccurs="0" maxOccurs="unbounded"/>
1770   </xs:sequence>
1771   <xs:attributeGroup ref="OperationNameGroup" />
1772   <xs:anyAttribute namespace="##other" processContents="lax"/>
1773 </xs:complexType>
1774
1775 <xs:element name="OperationMetrics" type="OperationMetricType"/>
1776
1777 <!-- MOWS::OperationalState -->
1778 <xs:complexType name="OperationalStateType">
1779   <xs:complexContent>
1780     <xs:extension base="muws2:StateType"/>
1781   </xs:complexContent>
1782 </xs:complexType>
1783 <xs:element name="UpState">
1784   <xs:complexType>
1785     <xs:complexContent>
1786       <xs:restriction base="mows:OperationalStateType"/>
1787     </xs:complexContent>
1788   </xs:complexType>
1789 </xs:element>
1790 <xs:element name="IdleState">
1791   <xs:complexType>
1792     <xs:complexContent>
1793       <xs:restriction base="mows:OperationalStateType">
1794         <xs:sequence>
1795           <xs:element ref="mows:UpState"/>
1796         </xs:sequence>
1797       </xs:restriction>
1798     </xs:complexContent>
1799   </xs:complexType>
1800 </xs:element>
1801 <xs:element name="BusyState">
1802   <xs:complexType>
1803     <xs:complexContent>
1804       <xs:restriction base="mows:OperationalStateType">
1805         <xs:sequence>
1806           <xs:element ref="mows:UpState"/>
1807         </xs:sequence>
1808       </xs:restriction>
1809     </xs:complexContent>

```

```

1810         </xs:complexType>
1811     </xs:element>
1812     <xs:element name="DownState">
1813         <xs:complexType>
1814             <xs:complexContent>
1815                 <xs:restriction base="mows:OperationalStateType"/>
1816             </xs:complexContent>
1817         </xs:complexType>
1818     </xs:element>
1819     <xs:element name="StoppedState">
1820         <xs:complexType>
1821             <xs:complexContent>
1822                 <xs:restriction base="mows:OperationalStateType">
1823                     <xs:sequence>
1824                         <xs:element ref="mows:DownState"/>
1825                     </xs:sequence>
1826                 </xs:restriction>
1827             </xs:complexContent>
1828         </xs:complexType>
1829     </xs:element>
1830     <xs:element name="CrashedState">
1831         <xs:complexType>
1832             <xs:complexContent>
1833                 <xs:restriction base="mows:OperationalStateType">
1834                     <xs:sequence>
1835                         <xs:element ref="mows:DownState"/>
1836                     </xs:sequence>
1837                 </xs:restriction>
1838             </xs:complexContent>
1839         </xs:complexType>
1840     </xs:element>
1841     <xs:element name="SaturatedState">
1842         <xs:complexType>
1843             <xs:complexContent>
1844                 <xs:restriction base="mows:OperationalStateType">
1845                     <xs:sequence>
1846                         <xs:element ref="mows:DownState"/>
1847                     </xs:sequence>
1848                 </xs:restriction>
1849             </xs:complexContent>
1850         </xs:complexType>
1851     </xs:element>
1852
1853     <xs:element name="CurrentOperationalState" type="mows:OperationalStateType"/>
1854     <xs:element name="LastOperationalStateTransition"
1855         type="muws2:StateTransitionType"/>
1856
1857     <!-- MOWS::RequestProcessingState -->
1858     <xs:complexType name="RequestProcessingStateType">
1859         <xs:complexContent>
1860             <xs:extension base="muws2:StateType"/>
1861         </xs:complexContent>
1862     </xs:complexType>
1863     <xs:element name="RequestReceivedState">
1864         <xs:complexType>

```

```

1865         <xs:complexContent>
1866             <xs:restriction base="mows:RequestProcessingStateType"/>
1867         </xs:complexContent>
1868     </xs:complexType>
1869 </xs:element>
1870 <xs:element name="RequestProcessingState">
1871     <xs:complexType>
1872         <xs:complexContent>
1873             <xs:restriction base="mows:RequestProcessingStateType"/>
1874         </xs:complexContent>
1875     </xs:complexType>
1876 </xs:element>
1877 <xs:element name="RequestCompletedState">
1878     <xs:complexType>
1879         <xs:complexContent>
1880             <xs:restriction base="mows:RequestProcessingStateType"/>
1881         </xs:complexContent>
1882     </xs:complexType>
1883 </xs:element>
1884 <xs:element name="RequestFailedState">
1885     <xs:complexType>
1886         <xs:complexContent>
1887             <xs:restriction base="mows:RequestProcessingStateType"/>
1888         </xs:complexContent>
1889     </xs:complexType>
1890 </xs:element>
1891 <xs:complexType name="MessageContentNotIncludedFlag"/>
1892 <xs:simpleType name="MessageSizeUnitType">
1893     <xs:restriction base="xs:string">
1894         <xs:enumeration value="bit"/>
1895         <xs:enumeration value="byte"/>
1896         <xs:enumeration value="word"/>
1897         <xs:enumeration value="dword"/>
1898         <xs:enumeration value="qword"/>
1899     </xs:restriction>
1900 </xs:simpleType>
1901 <xs:complexType name="MessageContentSizeType">
1902     <xs:simpleContent>
1903         <xs:extension base="xs:positiveInteger">
1904             <xs:attribute name="Unit"
1905                 type="mows:MessageSizeUnitType" use="required"/>
1906             <xs:anyAttribute namespace="##other" processContents="lax"/>
1907         </xs:extension>
1908     </xs:simpleContent>
1909 </xs:complexType>
1910 <xs:complexType name="MessageContentType">
1911     <xs:sequence>
1912         <xs:element name="Size"
1913             type="mows:MessageContentSizeType" minOccurs="0"/>
1914         <xs:choice>
1915             <xs:element name="NotIncluded"
1916                 type="mows:MessageContentNotIncludedFlag"/>
1917             <xs:element name="Text" type="xs:string"/>
1918             <xs:element name="Binary" type="xs:base64Binary"/>
1919             <xs:element name="Xml"

```

```

1920         type="mows:AnyXmlContentsType"/>
1921     </xs:choice>
1922     <xs:any namespace="##other" processContents="lax"
1923         minOccurs="0" maxOccurs="unbounded"/>
1924 </xs:sequence>
1925     <xs:anyAttribute namespace="##other" processContents="lax"/>
1926 </xs:complexType>
1927 <xs:complexType name="AnyXmlContentsType">
1928     <xs:sequence>
1929         <xs:any namespace="##any" processContents="lax"
1930             minOccurs="0" maxOccurs="unbounded"/>
1931     </xs:sequence>
1932     <xs:anyAttribute namespace="##any" processContents="lax"/>
1933 </xs:complexType>
1934 <xs:complexType name="MessageInformationType">
1935     <xs:sequence>
1936         <xs:element name="TransportInformation"
1937             type="mows:AnyXmlContentsType" minOccurs="0"/>
1938         <xs:element name="Message" type="mows:MessageContentType"/>
1939         <xs:any namespace="##any" processContents="lax"
1940             minOccurs="0" maxOccurs="unbounded"/>
1941     </xs:sequence>
1942     <xs:anyAttribute namespace="##any" processContents="lax"/>
1943 </xs:complexType>
1944 <xs:complexType name="RequestProcessingStateInformationType">
1945     <xs:sequence>
1946         <xs:element ref="muws2:StateTransition" maxOccurs="unbounded"/>
1947     </xs:sequence>
1948 </xs:complexType>
1949 <xs:element name="RequestProcessingNotification">
1950     <xs:complexType>
1951         <xs:sequence>
1952             <xs:element name="Request"
1953                 type="mows:MessageInformationType"
1954                 minOccurs="0"/>
1955             <xs:element name="Reply"
1956                 type="mows:MessageInformationType"
1957                 minOccurs="0"/>
1958             <xs:element name="StateInformation"
1959                 type="mows:RequestProcessingStateInformationType"/>
1960             <xs:any namespace="##any" processContents="lax"
1961                 minOccurs="0" maxOccurs="unbounded"/>
1962         </xs:sequence>
1963         <xs:attribute name="CurrentTime" type="xs:dateTime" use="required"/>
1964         <xs:anyAttribute namespace="##any" processContents="lax"/>
1965     </xs:complexType>
1966 </xs:element>
1967 <xs:simpleType name="IPv4AddressType">
1968     <xs:restriction base="xs:hexBinary">
1969         <xs:length value="8" fixed="true"/>
1970     </xs:restriction>
1971 </xs:simpleType>
1972 <xs:element name="IPv4Address" type="mows:IPv4AddressType"/>
1973 <xs:simpleType name="IPv6AddressType">
1974     <xs:restriction base="xs:hexBinary">

```

```

1975         <xs:length value="32" fixed="true"/>
1976     </xs:restriction>
1977 </xs:simpleType>
1978 <xs:element name="IPv6Address" type="mows:IPv6AddressType"/>
1979 <xs:simpleType name="TcplpDirectionType">
1980     <xs:restriction base="xs:string">
1981         <xs:enumeration value="to"/>
1982         <xs:enumeration value="from"/>
1983     </xs:restriction>
1984 </xs:simpleType>
1985 <xs:simpleType name="TcplpProtocolType">
1986     <xs:restriction base="xs:string">
1987         <xs:enumeration value="TCP"/>
1988         <xs:enumeration value="UDP"/>
1989     </xs:restriction>
1990 </xs:simpleType>
1991 <xs:element name="TcplpInfo">
1992     <xs:complexType>
1993         <xs:sequence>
1994             <xs:choice>
1995                 <xs:element ref="mows:IPv4Address"/>
1996                 <xs:element ref="mows:IPv6Address"/>
1997             </xs:choice>
1998             <xs:any namespace="##any" processContents="lax"
1999                 minOccurs="0" maxOccurs="unbounded"/>
2000         </xs:sequence>
2001         <xs:attribute name="Direction" type="mows:TcplpDirectionType"
2002             use="required"/>
2003         <xs:attribute name="Port" type="xs:positiveInteger" use="required"/>
2004         <xs:attribute name="Protocol" type="mows:TcplpProtocolType"
2005             use="required"/>
2006         <xs:anyAttribute namespace="##any" processContents="lax"/>
2007     </xs:complexType>
2008 </xs:element>
2009 </xs:schema>
2010
2011 <!--
2012         SCHEMA COPY Material
2013 Copy and paste element references below into the schema of a resource properties document.
2014 These references are provide to insure that the correct minOccurs/maxOccurs attributes are
2015 specified in a resource property document schema.
2016
2017 NOTE: You must import the MOWS schema namespace (mows).
2018
2019     ** Endpoint Identification Properties **
2020     <xs:element ref="mows:EndpointReference"/>
2021     <xs:element ref="mows:EndpointDescriptions" minOccurs="0"/>
2022
2023     ** MOWS Metric Properties **
2024     <xs:element ref="mows:NumberOfRequests" minOccurs="0"/>
2025     <xs:element ref="mows:NumberOfFailedRequests" minOccurs="0"/>
2026     <xs:element ref="mows:NumberOfSuccessfulRequests" minOccurs="0"/>
2027     <xs:element ref="mows:ServiceTime" minOccurs="0"/>
2028     <xs:element ref="mows:MaxResponseTime" minOccurs="0"/>
2029     <xs:element ref="mows:LastResponseTime" minOccurs="0"/>

```

```
2030 <xs:element ref="mows:MaxRequestSize" minOccurs="0"/>
2031 <xs:element ref="mows:LastRequestSize" minOccurs="0"/>
2032 <xs:element ref="mows:MaxResponseSize" minOccurs="0"/>
2033 <xs:element ref="mows:LastResponseSize" minOccurs="0"/>
2034
2035 **- MOWS Operation Metric Property **
2036 <xs:element ref="mows:OperationMetrics" minOccurs="0" maxOccurs="unbounded"/>
2037
2038 ** MOWS Operation Operational Status Property **
2039 <xs:element ref="mows:OperationOperationalStatus"
2040             minOccurs="0" maxOccurs="unbounded"/>
2041
2042 ** Operational State Properties **
2043 <xs:element ref="mows:CurrentOperationalState"/>
2044 <xs:element ref="mows:LastOperationalStateTransition" minOccurs="0"/>
2045
2046 -->
2047
```

2048

Appendix E. WSDL elements

```
2049 <?xml version="1.0" encoding="utf-8"?>
2050 <w:definitions xmlns:w="http://schemas.xmlsoap.org/wsdl/"
2051     xmlns:xs="http://www.w3.org/2001/XMLSchema"
2052     xmlns:wsrp="http://docs.oasis-open.org/wsrp/rp-2"
2053     xmlns:mows="http://docs.oasis-open.org/wsdm/mows-2.xsd"
2054     xmlns:mowsw="http://docs.oasis-open.org/wsdm/mows-2.wsdl"
2055     targetNamespace="http://docs.oasis-open.org/wsdm/mows-2.wsdl">
2056
2057     <w:types>
2058         <xs:import namespace="http://docs.oasis-open.org/wsdm/mows-2.xsd"
2059 schemaLocation="http://docs.oasis-open.org/wsdm/mows-2.xsd"/>
2060     </w:types>
2061
2062     <w:message name="GetManageabilityReferencesRequest">
2063         <w:part name="body" element="mows:GetManageabilityReferences"/>
2064     </w:message>
2065     <w:message name="GetManageabilityReferencesResponse">
2066         <w:part name="body"
2067             element="mows:GetManageabilityReferencesResponse"/>
2068     </w:message>
2069
2070 </w:definitions>
2071
2072 <!--
2073         WSDL COPY Material
2074 Cut and Paste the operation specification below into a portType definition of the WSDL
2075 documents of a web service.
2076
2077 NOTE: You must import the MOWS WSDL (mowsw).
2078
2079     <operation name="GetManageabilityReferences">
2080         <input name="GetManageabilityReferencesRequest"
2081             message="mowsw:GetManageabilityReferencesRequest"/>
2082         <output name="GetManageabilityReferencesResponse"
2083             message="mowsw:GetManageabilityReferencesResponse"/>
2084     </operation>
2085
2086 -->
2087
```

Appendix F. Notification topic spaces

```

2089 <?xml version="1.0" encoding="utf-8"?>
2090 <wstop:TopicSpace name="MOWS"
2091     targetNamespace="http://docs.oasis-open.org/wsdm/mowse-2.xml"
2092     xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
2093     xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
2094     xmlns:mows="http://docs.oasis-open.org/wsdm/mows-2.xsd"
2095     xmlns:wstop="http://docs.oasis-open.org/wsn/t-1">
2096
2097 <wstop:Topic name="IdentificationCapability" messageTypes="muws1:ManagementEvent"/>
2098 <wstop:Topic name="MetricsCapability" messageTypes="muws1:ManagementEvent"/>
2099 <wstop:Topic name="OperationMetricsCapability" messageTypes="muws1:ManagementEvent"/>
2100 <wstop:Topic name="OperationalStateCapability" messageTypes="muws1:ManagementEvent"/>
2101 <wstop:Topic name="OperationalStatusCapability"
2102     messageTypes="muws1:ManagementEvent"/>
2103 <wstop:Topic name="OperationOperationalStatusCapability"
2104     messageTypes="muws1:ManagementEvent"/>
2105 <wstop:Topic name="RequestProcessingStateCapability"
2106     messageTypes="muws1:ManagementEvent"/>
2107
2108
2109 <wstop:Topic name="RequestProcessingObservations"
2110     messageTypes="muws1:ManagementEvent">
2111     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2112 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2113 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2114     </wstop:MessagePattern>
2115     <wstop:Topic name="RequestReceived"
2116         messageTypes="muws1:ManagementEvent">
2117         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2118 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2119 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2120         </wstop:MessagePattern>
2121         </wstop:Topic>
2122     <wstop:Topic name="RequestProcessing"
2123         messageTypes="muws1:ManagementEvent">
2124         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2125 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2126 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2127         </wstop:MessagePattern>
2128         </wstop:Topic>
2129     <wstop:Topic name="RequestCompleted"
2130         messageTypes="muws1:ManagementEvent">
2131         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2132 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2133 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2134         </wstop:MessagePattern>
2135         </wstop:Topic>
2136     <wstop:Topic name="RequestFailed"
2137         messageTypes="muws1:ManagementEvent">
2138     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">

```



```

2139 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2140 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2141     </wstop:MessagePattern>
2142     </wstop:Topic>
2143     <wstop:Topic name="Digest"
2144         messageTypes="muws1:ManagementEvent">
2145         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2146 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2147 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2148     </wstop:MessagePattern>
2149     </wstop:Topic>
2150 </wstop:Topic>
2151
2152 <wstop:Topic name="RequestProcessingObservationsWithAttachments"
2153     messageTypes="muws1:ManagementEvent">
2154     <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2155 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2156 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2157     </wstop:MessagePattern>
2158     <wstop:Topic name="RequestReceived"
2159         messageTypes="muws1:ManagementEvent">
2160         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2161 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2162 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2163     </wstop:MessagePattern>
2164     </wstop:Topic>
2165     <wstop:Topic name="RequestProcessing"
2166         messageTypes="muws1:ManagementEvent">
2167         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2168 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2169 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2170     </wstop:MessagePattern>
2171     </wstop:Topic>
2172     <wstop:Topic name="RequestCompleted"
2173         messageTypes="muws1:ManagementEvent">
2174         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2175 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2176 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2177     </wstop:MessagePattern>
2178     </wstop:Topic>
2179     <wstop:Topic name="RequestFailed"
2180         messageTypes="muws1:ManagementEvent">
2181         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2182 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2183 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2184     </wstop:MessagePattern>
2185     </wstop:Topic>
2186     <wstop:Topic name="Digest"
2187         messageTypes="muws1:ManagementEvent">
2188         <wstop:MessagePattern Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
2189 //muws1:ManagementEvent[muws2:Situation/muws2:SituationCategory//muws2:ReportSituation
2190 and muws2:Severity="1" and count(mows:RequestProcessingNotification)=1]
2191     </wstop:MessagePattern>
2192     </wstop:Topic>
2193 </wstop:Topic>

```

