



1
2
3
4

Document Number: DSP0226

Date: 2014-06-16

Version: 1.2.0b

5 **Web Services for Management (WS-**
6 **Management) Specification**

7 **Document Type: Specification**

8 **Document Status: Work in Progress**

9 **Document Language: en-US**

10 Copyright Notice

11 Copyright © 2006–2014 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

12 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
13 management and interoperability. Members and non-members may reproduce DMTF specifications and
14 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
15 time, the particular version and release date should always be noted.

16 Implementation of certain elements of this standard or proposed standard may be subject to third party
17 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
18 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
19 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
20 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
21 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
22 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
23 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
24 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
25 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
26 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
27 implementing the standard from any and all claims of infringement by a patent owner for such
28 implementations.

29 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
30 such patent may relate to or impact implementations of DMTF standards, visit
31 <http://www.dmtf.org/about/policies/disclosures.php>.

32

CONTENTS

34	Foreword.....	7
35	1 Scope.....	9
36	2 Normative References	9
37	3 Terms and Definitions	11
38	4 Symbols and Abbreviated Terms	14
39	5 Addressing.....	15
40	5.1 Management Addressing	15
41	5.2 Versions of Addressing.....	23
42	5.3 Requirements for Compatibility	24
43	5.4 Use of Addressing in WS-Management.....	26
44	6 WS-Management Control Headers.....	43
45	6.1 wsman:OperationTimeout	43
46	6.2 wsman:MaxEnvelopeSize	44
47	6.3 wsman:Locale	45
48	6.4 wsman:OptionSet	46
49	6.5 wsman:RequestEPR	49
50	7 Resource Access.....	50
51	7.1 General.....	50
52	7.2 Addressing Uniformity	52
53	7.3 Get.....	53
54	7.4 Put	55
55	7.5 Delete	58
56	7.6 Create.....	60
57	7.7 Fragment-Level Access.....	63
58	7.8 Fragment-Level Get	65
59	7.9 Fragment-Level Put.....	67
60	7.10 Fragment-Level Delete.....	69
61	7.11 Fragment-Level Create	70
62	8 Enumeration of Datasets	72
63	8.1 General.....	72
64	8.2 Enumerate.....	74
65	8.3 Filter Interpretation	81
66	8.4 Pull.....	83
67	8.5 Release	87
68	8.6 Ad-Hoc Queries and Fragment-Level Enumerations	89
69	8.7 Enumeration of EPRs.....	90
70	8.8 Renew	91
71	8.9 GetStatus.....	93
72	8.10 EnumerationEnd.....	93
73	9 Custom Actions (Methods).....	94
74	10 Notifications (Eventing).....	95
75	10.1 General.....	95
76	10.2 Subscribe.....	96
77	10.3 GetStatus.....	116
78	10.4 Unsubscribe.....	117
79	10.5 Renew	118
80	10.6 SubscriptionEnd	119
81	10.7 Acknowledgement of Delivery	121
82	10.8 Refusal of Delivery	122
83	10.9 Dropped Events.....	123
84	10.10 Access Control	124

85	10.11 Implementation Considerations.....	125
86	10.12 Advertisement of Notifications.....	125
87	11 Metadata and Discovery.....	125
88	12 Security.....	128
89	12.1 General.....	128
90	12.2 Security Profiles.....	129
91	12.3 Security Considerations for Event Subscriptions.....	129
92	12.4 Including Credentials with a Subscription.....	131
93	12.5 Correlating Events with a Subscription.....	131
94	12.6 Transport-Level Authentication Failure.....	131
95	12.7 Security Implications of Third-Party Subscriptions.....	131
96	13 Transports and Message Encoding.....	132
97	13.1 SOAP.....	132
98	13.2 Lack of Response.....	133
99	13.3 Replay of Messages.....	133
100	13.4 Encoding Limits.....	133
101	13.5 Binary Attachments.....	134
102	13.6 Case-Sensitivity.....	134
103	14 Faults.....	135
104	14.1 Introduction.....	135
105	14.2 Fault Encoding.....	135
106	14.3 NotUnderstood Faults.....	136
107	14.4 Degenerate Faults.....	137
108	14.5 Fault Extensibility.....	137
109	14.6 Master Faults.....	138
110	ANNEX A (informative) Notational Conventions.....	158
111	A.1 XML Namespaces.....	158
112	ANNEX B (normative) Conformance.....	160
113	ANNEX C (normative) HTTP(S) Transport and Security Profile.....	161
114	C.1 General.....	161
115	C.2 HTTP(S) Binding.....	161
116	C.3 HTTP(S) Security Profiles.....	162
117	C.4 IPSec and HTTP.....	167
118	ANNEX D (informative) XPath Support.....	169
119	D.1 General.....	169
120	D.2 Level 1.....	170
121	D.3 Level 2.....	172
122	ANNEX E (normative) Selector Filter Dialect.....	175
123	ANNEX F (informative) Identify XML Schema.....	177
124	ANNEX G (informative) Resource Access Operations XML Schema and WSDL.....	180
125	ANNEX H (informative) Enumeration Operations XML Schema and WSDL.....	185
126	ANNEX I (informative) Notification OperationsXML Schema and WSDL.....	194
127	ANNEX J (informative) Addressing XML Schema.....	202
128	ANNEX K (informative) WS-Management XML Schema.....	205
129	ANNEX L (informative) Change Log.....	215
130		

131 **Figures**

132	Figure 1 – Message Information Header Blocks	19
-----	--	----

133

134 **Tables**

135	Table 1 – Relationship Type	20
136	Table 2 – Interoperability Requirements	24
137	Table 3 – WSA Versions in Exchanges	25
138	Table 4 – wsa:Action URI Descriptions	41
139	Table 5 – wsman:AccessDenied	138
140	Table 6 – wsa:ActionNotSupported	139
141	Table 7 – wsman:AlreadyExists	139
142	Table 8 – wsmen:CannotProcessFilter	140
143	Table 9 – wsman:CannotProcessFilter	140
144	Table 10 – wsman:Concurrency	141
145	Table 11 – wsme:DeliveryModeRequestedUnavailable	141
146	Table 12 – wsman:DeliveryRefused	142
147	Table 13 – wsa:DestinationUnreachable	142
148	Table 14 – wsman:EncodingLimit	143
149	Table 15 – wsa:EndpointUnavailable	143
150	Table 16 – wsman:EventDeliverToUnusable	144
151	Table 17 – wsme:EventSourceUnableToProcess	145
152	Table 18 – wsmen:FilterDialectRequestedUnavailable	145
153	Table 19 – wsme:FilteringNotSupported	145
154	Table 20 – wsmen:FilteringNotSupported	146
155	Table 21 – wsme:FilteringRequestedUnavailable	146
156	Table 22 – wsman:FragmentDialectNotSupported	147
157	Table 23 – wsman:InternalError	147
158	Table 24 – wsman:InvalidBookmark	148
159	Table 25 – wsmen:InvalidEnumerationContext	148
160	Table 26 – wsme:InvalidExpirationTime	149
161	Table 27 – wsmen:InvalidExpirationTime	149
162	Table 28 – wsme:InvalidMessage	150
163	Table 29 – wsa:InvalidMessageInformationHeader	150
164	Table 30 – wsman:InvalidOptions	151
165	Table 31 – wsman:InvalidParameter	151
166	Table 32 – wsmt:InvalidRepresentation	151
167	Table 33 – wsman:InvalidSelectors	152
168	Table 34 – wsa:MessageInformationHeaderRequired	153
169	Table 35 – wsman:NoAck	153
170	Table 36 – wsman:QuotaLimit	153
171	Table 37 – wsman:SchemaValidationError	154

172 Table 38 – wsmen:TimedOut 154

173 Table 39 – wsman:TimedOut 154

174 Table 40 – wsme:UnableToRenew 155

175 Table 41 – wsme:UnsupportedExpirationType 155

176 Table 42 – wsmen:UnsupportedExpirationType 155

177 Table 43 – wsman:UnsupportedFeature 156

178 Table 44 – wsme:UnsupportedExpirationType 156

179 Table 45 – wsmen:UnableToRenew 157

180 Table 46 – wsa:InvalidMessage 157

181 Table 47 – wsme:CannotProcessFilter 157

182 Table A-1 – Prefixes and XML Namespaces Used in This Specification 159

183 Table C-1 – Basic Authentication Sequence 163

184 Table C-2 – Digest Authentication Sequence 164

185 Table C-3 – Basic Authentication over HTTPS Sequence 164

186 Table C-4 – Digest Authentication over HTTPS Sequence 164

187 Table C-5 – HTTPS with Client Certificate Sequence 165

188 Table C-6 – Basic Authentication over HTTPS with Client Certificate Sequence 165

189 Table C-7 – SPNEGO Authentication over HTTPS Sequence 166

190 Table C-8 – SPNEGO Authentication over HTTPS with Client Certificate Sequence 167

191 Table D-1 – XPath Level 1 Terminals 171

192 Table D-2 – XPath Level 2 Terminals 173

193

194

Foreword

195 The *Web Services for Management (WS-Management) Specification* (DSP0226) was prepared by the
196 WS-Management sub-group of the WBEM Infrastructure & Protocols Working Group.

197 This International Standard makes use of functionality similar to the following W3C
198 Recommendations:

- 199 • Web Services Eventing (WS-Eventing)
- 200 • Web Services Transfer (WS-Transfer)
- 201 • Web Services Enumeration (WS-Enumeration)

202 These W3C Recommendations were not available at the time WS-Management was defined, and
203 similar functionality was incorporated directly into provisions of the WS-Management specification.
204 Future revisions of WS-Management might incorporate these functions by External Reference to
205 these W3C Recommendations

206 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and
207 systems management and interoperability.

208 Acknowledgements

209 The authors wish to acknowledge the following people.

210 Editors:

- 211 • Keith Bankston - Microsoft
- 212 • Larry Lamers - VMware

213 Authors:

- 214 • Akhil Arora – Sun Microsystems
- 215 • Vince Brunssen – IBM
- 216 • Nathan Burkhart – Microsoft
- 217 • Mark Carlson – Sun Microsystems
- 218 • Josh Cohen – Microsoft
- 219 • Doug Davis – IBM
- 220 • Jim Davis – WBEM Solutions
- 221 • Tony Dicenzo – Oracle
- 222 • Mike Dutch – Symantec
- 223 • Zulah Eckert – BEA Systems
- 224 • George Ericson – EMC
- 225 • Wassim Fayed – Microsoft
- 226 • Chris Ferris – IBM
- 227 • Bob Freund – Hitachi Ltd.
- 228 • Eugene Golovinsky – BMC Software
- 229 • Yasuhiro Hagiwara – NEC
- 230 • Steve Hand – Olocity
- 231 • Jackson He – Intel
- 232 • David Hines – Intel
- 233 • Reiji Inohara – NEC
- 234 • Christane Kämpfe – Fujitsu-Siemens Computers
- 235 • Paul Knight – Nortel Networks
- 236 • Vincent Kowalski – BMC Software
- 237 • Heather Kreger – IBM

- 238 • Vishwa Kumbalimutt – Microsoft
- 239 • Sunil Kunisetty – Oracle
- 240 • Richard Landau – Dell
- 241 • Paul Lipton – CA
- 242 • James Martin – Intel
- 243 • Raymond McCollum – Microsoft
- 244 • Milan Milenkovic – Intel
- 245 • Jeff Mischkinsky – Oracle
- 246 • Paul Montgomery – AMD
- 247 • Jishnu Mukurji – HP
- 248 • Bryan Murray – HP
- 249 • Alexander Nosov – Microsoft
- 250 • Abhay Padlia – Novell
- 251 • Gilbert Pilz – Oracle
- 252 • Roger Reich – Symantec
- 253 • Brian Reistad – Microsoft
- 254 • Larry Russon – Novell
- 255 • Tom Rutt – Fujitsu Ltd.
- 256 • Jeffrey Schlimmer – Microsoft
- 257 • Dr. Hemal Shah – Broadcom
- 258 • Sharon Smith – Intel
- 259 • Enoch Suen – Dell
- 260 • Vijay Tewari – Intel
- 261 • William Vambenepe – HP
- 262 • Andrea Westerinen – CA, Inc.
- 263 • Kirk Wilson – CA, Inc.
- 264 • Dr. Jerry Xie – Intel

Contributors:

- 266 • Paul C. Allen – Microsoft
- 267 • Rodrigo Bomfim – Microsoft
- 268 • Don Box – Microsoft
- 269 • Jerry Duke – Intel
- 270 • David Filani – Intel
- 271 • Kirill Gavrylyuk – Microsoft
- 272 • Omri Gazitt – Microsoft
- 273 • Frank Gorishek – AMD
- 274 • Lawson Guthrie – Intel
- 275 • Arvind Kumar – Intel
- 276 • Brad Lovering – Microsoft
- 277 • Pat Maynard – Intel
- 278 • Steve Millet – Microsoft
- 279 • Matthew Senft – Microsoft
- 280 • Barry Shilmover – Microsoft
- 281 • Tom Slight – Intel
- 282 • Marvin Theimer – Microsoft
- 283 • Dave Tobias – AMD
- 284 • John Tollefsrud – Sun
- 285 • Anders Vinberg – Microsoft
- 286 • Megan Wallent – Microsoft
- 287 •

288
289

Web Services for Management (WS-Management) Specification

290 1 Scope

291 The *Web Services for Management (WS-Management) Specification* describes a Web services
292 protocol based on SOAP for use in management-specific domains. These domains include the
293 management of entities such as PCs, servers, devices, Web services and other applications, and
294 other manageable entities. Services can expose only a WS-Management interface or compose the
295 WS-Management service interface with some of the many other Web service specifications.

296 A crucial application for these services is in the area of systems management. To promote
297 interoperability between management applications and managed resources, this specification
298 identifies a core set of Web service specifications and usage requirements that expose a common set
299 of operations central to all systems management. This includes the ability to do the following:

- 300 • Get, put (update), create, and delete individual resource instances, such as settings and
301 dynamic values
- 302 • Enumerate the contents of containers and collections, such as large tables and logs
- 303 • Subscribe to events emitted by managed resources
- 304 • Execute specific management methods with strongly typed input and output parameters

305 In each of these areas of scope, this specification defines minimal implementation requirements for
306 conformant Web service implementations. An implementation is free to extend beyond this set of
307 operations, and to choose not to support one or more of the preceding areas of functionality if that
308 functionality is not appropriate to the target device or system.

309 This specification intends to meet the following requirements:

- 310 • Constrain Web services protocols and formats so that Web services can be implemented
311 with a small footprint in both hardware and software management services.
- 312 • Define minimum requirements for compliance without constraining richer implementations.
- 313 • Ensure backward compatibility and interoperability with WS-Management version 1.0 and
314 1.1.
- 315 • Ensure composability with other Web services specifications.

316 2 Normative References

317 The following referenced documents are indispensable for the application of this document. For dated
318 references, only the edition cited applies. For undated references, the latest edition of the referenced
319 document (including any amendments) applies.

320 IETF RFC 2616, R. Fielding et al, *Hypertext Transfer Protocol (HTTP 1.1)*, June 1999,
321 <http://tools.ietf.org/html/rfc2616>

322 IETF, RFC 3986, T. Berners-Lee et al, *Uniform Resource Identifiers (URI): Generic Syntax*, August
323 1998, <http://tools.ietf.org/html/rfc3986>

324 IETF, RFC 4122, P. Leach et al, *A Universally Unique Identifier (UUID) URN Namespace*, July 2005,
325 <http://tools.ietf.org/html/rfc4122>

- 326 IETF RFC 4178, L. Zhu et al, *The Simple and Protected Generic Security Service Application*
327 *Program Interface (GSS-API) Negotiation Mechanism*, October 2005, <http://tools.ietf.org/html/rfc4178>
- 328 IETF, RFC 4559, K. Jaganathan et al, *SPNEGO-based Kerberos and NTLM HTTP Authentication in*
329 *Microsoft Windows*, June 2006, <http://www.ietf.org/rfc/rfc4559.txt>
- 330 IETF RFC 5646, A. Phillips et al, *Tags for Identifying Languages*, September 2009,
331 <http://tools.ietf.org/rfc/rfc5646.txt>
- 332 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
333 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>
- 334 OASIS, A. Nadalin et al, *Web Services Security Username Token Profile 1.0*, March 2004,
335 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf>
- 336 OASIS, A. Nadalin et al, *Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)*,
337 March 2004, <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>
338
- 339 OASIS, S. Anderson et al, *Web Services Trust Language (WS-Trust)*, December 2005,
340 <http://schemas.xmlsoap.org/ws/2005/02/trust>
- 341 The Unicode Consortium, *The Unicode Standard Version 3.0*, January 2000,
342 <http://www.unicode.org/book/u2.html>
- 343 The Unicode Consortium, *Byte Order Mark (BOM) FAQ*,
344 http://www.unicode.org/fag/utf_bom.html#BOM
- 345 W3C, M. Gudgin, et al, *SOAP Version 1.2 Part 1: Messaging Framework*, June 2003,
346 <http://www.w3.org/TR/soap12-part1/>
- 347 W3C, M. Gudgin, et al, *SOAP Version 1.2 Part 2: Adjuncts*, June 2003,
348 <http://www.w3.org/TR/2003/REC-soap12-part2-20030624>
- 349 W3C, M. Gudgin, et al, *SOAP Message Transmission Optimization Mechanism (MTOM)*,
350 November 2004, <http://www.w3.org/TR/2004/PR-soap12-mtom-20041116/>
- 351 W3C, J. Clark et al, *XML Path Language Version 1.0 (XPath 1.0)*, November 1999,
352 <http://www.w3.org/TR/1999/REC-xpath-19991116>
- 353 W3C, J. Cowan et al, *XML Information Set Second Edition (XML Infoset)*, February 2004,
354 <http://www.w3.org/TR/2004/REC-xml-infoset-20040204/>
- 355 W3C, H. Thompson et al, *XML Schema Part 1: Structures (XML Schema 1)*, October 2004,
356 <http://www.w3.org/TR/xmlschema-1/>
- 357 W3C, P. Biron et al, *XML Schema Part 2: Datatypes (XML Schema 2)*, October 2004,
358 <http://www.w3.org/TR/xmlschema-2/>
- 359 ISO/IEC 40240:2011 Information technology -- W3C Web Services Addressing 1.0 – Core,
360 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=58365
- 361 ISO/IEC 40250:2011 Information technology -- W3C Web Services Addressing 1.0 -- SOAP Binding,
362 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=58375
- 363 ISO/IEC 40260:2011 Information technology -- W3C Web Services Addressing 1.0 – Metadata,
364 http://www.iso.org/iso/catalogue_detail?csnumber=58385 W3C, *Extensible Markup Language*
365 *(XML) 1.0*, W3C Recommendation, October 2000, <http://www.w3.org/TR/2000/REC-xml-20001006>
- 366 W3C, *Namespaces in XML*, W3C Recommendation, January 1999,
367 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>
- 368 W3C, E. Christensen et al, *Web Services Description Language Version 1.1 (WSDL/1.1)*, March
369 2001, <http://www.w3.org/TR/wsdl>

370 W3C, S. Boag et al, XQuery 1.0: An XML Query Language (XQuery 1.0), January 2007,
371 <http://www.w3.org/TR/2007/REC-xquery-20070123/>

372 **3 Terms and Definitions**

373 For the purposes of this document, the following terms and definitions apply. The fact that a
374 normative term such as "shall", "shall not", "should", "should not", "may", or "need not" may be used in
375 text which does not have an associated rule number does not mean that the text is not normative.

376 **3.1**

377 **can**

378 used for statements of possibility and capability, whether material, physical, or causal

379 **3.2**

380 **cannot**

381 used for statements of possibility and capability, whether material, physical, or causal

382 **3.3**

383 **conditional**

384 indicates requirements to be followed strictly to conform to the document when the specified
385 conditions are met

386 **3.4**

387 **mandatory**

388 indicates requirements to be followed strictly to conform to the document and from which no deviation
389 is permitted

390 **3.5**

391 **may**

392 indicates a course of action permissible within the limits of the document

393 **3.6**

394 **need not**

395 indicates a course of action permissible within the limits of the document

396 **3.7**

397 **optional**

398 indicates a course of action permissible within the limits of the document

399 **3.8**

400 **shall**

401 indicates requirements to be followed strictly to conform to the document and from which no deviation
402 is permitted

403 **3.9**

404 **shall not**

405 indicates requirements to be followed strictly to conform to the document and from which no deviation
406 is permitted

407 **3.10**

408 **should**

409 indicates that among several possibilities, one is recommended as particularly suitable, without
410 mentioning or excluding others, or that a certain course of action is preferred but not necessarily
411 required

- 412 **3.11**
413 **should not**
414 indicates that a certain possibility or course of action is deprecated but not prohibited
- 415 **3.12**
416 **client**
417 the application that uses the Web services defined in this document to access the management
418 service
- 419 **3.13**
420 **consumer**
421 the Web service that is requesting the data enumeration from the data source
- 422 **3.14**
423 **data source**
424 a Web service that supports traversal using enumeration contexts via the Enumerate operation
425 defined in this specification
- 426 **3.15**
427 **delivery mode**
428 the mechanism by which notification messages are delivered from the source to the sink
- 429 **3.16**
430 **enumeration context**
431 a session context that represents a specific traversal through a logical sequence of XML element
432 information items using the Pull operation defined in this specification
- 433 **3.17**
434 **event sink**
435 a Web service that receives notifications
- 436 **3.18**
437 **event source**
438 a Web service that sends notifications and accepts requests to create subscriptions
- 439 **3.19**
440 **managed resource**
441 an entity that can be of interest to an administrator
442 It may be a physical object, such as a laptop computer or a printer, or an abstract entity, such as a
443 service.
- 444 **3.20**
445 **notification**
446 a message sent to indicate that an event has occurred
- 447 **3.21**
448 **push mode**
449 a delivery mechanism where the source sends event messages to the sink as individual, unsolicited
450 SOAP messages

451 **3.22**

452 **resource**

453 a Web service that is addressable by an endpoint reference and accessed using the operations
454 defined in this specification. This resource can be represented by an XML document. The XML
455 document may be a representation of managed resource

456 **3.23**

457 **resource class**

458 an abstract representation (type) of a managed resource

459 A resource class defines the representation of management-related operations and properties. An
460 example of a resource class is the description of operations and properties for a set of laptop
461 computers.

462 **3.24**

463 **resource factory**

464 a Web service that is capable of creating new resources using the Create operation defined in this
465 specification

466 **3.25**

467 **resource instance**

468 an instantiation of a resource class

469 An example is the set of management-related operations and property values for a specific laptop
470 computer.

471 **3.26**

472 **selector**

473 a resource-relative name and value pair that acts as an instance-level discriminant when used with
474 the WS-Management default addressing model

475 A selector is essentially a filter or "key" that identifies the desired instance of the resource. A selector
476 may not be present when service-specific addressing models are used.

477 The relationship of services to resource classes and instances is as follows:

- 478 • A service consists of one or more resource classes.
- 479 • A resource class may contain zero or more instances.

480 If more than one instance for a resource class exists, they are isolated or identified through parts of
481 the SOAP address for the resource, such as the ResourceURI and SelectorSet fields in the default
482 addressing model.

483 **3.27**

484 **service**

485 an application that provides management services to clients by exposing the Web services defined in
486 this document

487 Typically, a service is equivalent to the network "listener," is associated with a physical transport
488 address, and is essentially a type of manageability access point.

489 **3.28**

490 **subscriber**

491 a Web service that sends requests to create, renew, and/or delete subscriptions

492 **3.29**

493 **subscription manager**

494 a Web service that accepts requests to manage, get the status of, renew, and/or delete subscriptions
495 on behalf of an event source

496 **4 Symbols and Abbreviated Terms**

497 The following symbols and abbreviations are used in this document.

498 **4.1**

499 **BNF**

500 Backus-Naur Form (<http://foldoc.org/foldoc/?Backus-Naur+Form>)

501 **4.2**

502 **BOM**

503 byte-order mark

504 **4.3**

505 **CQL**

506 CIM Query Language

507 **4.4**

508 **EPR**

509 Endpoint Reference

510 **4.5**

511 **GSSAPI**

512 Generic Security Services Application Program Interface

513 **4.6**

514 **SOAP**

515 Simple Object Access Protocol

516 **4.7**

517 **SPNEGO**

518 Simple and Protected GSSAPI Negotiation Mechanism

519 **4.8**

520 **SQL**

521 Structured Query Language

522 **4.9**

523 **URI**

524 Uniform Resource Identifier

525 **4.10**

526 **URL**

527 Uniform Resource Locator

528 **4.11**

529 **UTF**

530 UCS Transformation Format

531 **4.12**

532 **UUID**

533 Universally Unique Identifier

534 **4.13**

535 **WSDL**

536 Web Services Description Language

537 **4.14**

538 **WS-Man**

539 Web Services Management

540 **5 Addressing**

541 WS-Management relies on a SOAP-based addressing mechanism (like the one defined in 5.1) to
542 define references to other Web service endpoints and to define some of the headers used in SOAP
543 messages. This addressing mechanism is semantically equivalent and fully wire-compatible with the
544 version of WS-Addressing referenced in WS-Management 1.0. Therefore, this change to WS-
545 Management is fully backward compatible with existing WS-Management implementations.

546 Clause 5.2 specifies how more than one addressing version may be used with WS-Management,
547 such as the version defined in 5.1 or the W3C Recommendation version of addressing. In this
548 specification, unless explicitly referring to a particular version, the term "Addressing" refers generically
549 to either version of addressing as defined in 5.2.

550 Multiple addressing models may be used with any of the addressing versions described in 5.2.
551 Implementations may implement any of the following addressing models:

- 552 • basic addressing as defined in 5.1
- 553 • the Default Addressing Model as defined in 5.4.2
- 554 • new addressing models that are not defined in this specification. These addressing models
555 may impose additional restrictions or requirements for addressing.

556 **5.1 Management Addressing**

557 The features defined in this clause provide a transport-neutral mechanism to address Web services
558 and messages. Specifically, this clause defines XML elements to identify Web service endpoints and
559 to secure end-to-end endpoint identification in messages. This enables messaging systems to
560 support message transmission through networks that include processing nodes such as endpoint
561 managers, firewalls, and gateways in a transport-neutral manner.

562 **5.1.1 Introduction**

563 This clause defines two interoperable constructs, endpoint references and message information
564 headers, that convey information that is typically provided by transport protocols and messaging
565 systems. These constructs normalize this underlying information into a uniform format that can be
566 processed independently of transport or application.

567 A Web service endpoint is an entity, processor, or resource that can be referenced and can be
568 targeted for Web service messages. Endpoint references convey the information needed to identify
569 and reference a Web service endpoint, and they may be used in several different ways:

- 570 • Endpoint references are suitable for conveying the information needed to access a Web
571 service endpoint.
- 572 • Endpoint references are also used to provide addresses for individual messages sent to
573 and from Web services.

574 To deal with the latter use case, this clause defines a family of message information headers that
575 allows uniform addressing of messages independent of underlying transport. These message
576 information headers convey end-to-end message characteristics including addressing for source and
577 destination endpoints as well as message identity.

578 EXAMPLE: The following example illustrates the use of these mechanisms in a SOAP 1.2 message being sent
579 from <http://business456.example/client1> to <http://fabrikam123.example/Purchasing>.

580 Lines (002) to (014) represent the header of the SOAP message where the mechanisms defined in this clause
581 are used. The body is represented by lines (015) to (017).

582 Lines (003) to (013) contain the message information header blocks. Specifically, lines (003) to (005) specify the
583 identifier for this message, lines (006) to (008) specify the endpoint from where the message originated, and
584 lines (009) to (011) specify the endpoint to which replies to this message should be sent as an Endpoint
585 Reference. Line (012) specifies the address URI of the ultimate receiver of this message. Line (013) specifies an
586 Action URI identifying expected semantics.

```
587 (001) <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
588         xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
589 (002)   <S:Header>
590 (003)     <wsa:MessageID>
591 (004)       uuid:6B29FC40-CA47-1067-B31D-00DD010662DA
592 (005)     </wsa:MessageID>
593 (006)     <wsa:From>
594 (007)       <wsa:Address>http://business456.example/client1</wsa:Address>
595 (008)     </wsa:From>
596 (009)     <wsa:ReplyTo>
597 (010)       <wsa:Address>http://business456.example/client1</wsa:Address>
598 (011)     </wsa:ReplyTo>
599 (012)     <wsa:To>http://fabrikam123.example/Purchasing</wsa:To>
600 (013)     <wsa:Action>http://fabrikam123.example/SubmitPO</wsa:Action>
601 (014)   </S:Header>
602 (015)   <S:Body>
603 (016)     ...
604 (017)   </S:Body>
605 (018) </S:Envelope>
```

606 5.1.2 Endpoint References

607 This clause defines the syntax of an Endpoint Reference (EPR).

608 5.1.2.1 Format of Endpoint References

609 This clause defines an XML representation for an endpoint reference as both an XML type
610 (`wsa:EndpointReferenceType`) and as an XML element (`<wsa:EndpointReference>`).

611 The `wsa:EndpointReferenceType` type is used wherever a Web service endpoint is referenced. The
612 following describes the contents of this type:

```
613 <wsa:EndpointReference>
614   <wsa:Address>xs:anyURI</wsa:Address>
615   <wsa:ReferenceProperties>... </wsa:ReferenceProperties> ?
616   <wsa:ReferenceParameters>... </wsa:ReferenceParameters> ?
617   <wsa:PortType>xs:QName</wsa:PortType> ?
618   <wsa:ServiceName PortName="xs:NCName"?>xs:QName</wsa:ServiceName> ?
619   <wsp:Policy> ... </wsp:Policy>*
620 </wsa:EndpointReference>
```

621 The following describes the attributes and elements listed in the preceding schema overview:

622 `wsa:EndpointReference`

623 This represents some element of type `wsa:EndpointReferenceType`. This example uses the
624 predefined `<wsa:EndpointReference>` element, but any element of type
625 `wsa:EndpointReferenceType` may be used.

- 626 `wsa:EndpointReference/wsa:Address`
- 627 This required element (of type `xs:anyURI`) specifies the address URI that identifies the endpoint.
628 This address may be a logical address or identifier for the service endpoint.
- 629 `wsa:EndpointReference/wsa:ReferenceProperties/`
- 630 This optional element contains any number of individual reference properties that are associated
631 with the endpoint to facilitate a particular interaction. Reference properties are XML elements that
632 are required to properly interact with the endpoint. Reference properties are provided by the issuer
633 of the endpoint reference and are otherwise assumed to be opaque to consuming applications.
- 634 NOTE: The use of reference properties is deprecated; reference parameters should be used instead.
- 635 `wsa:EndpointReference/wsa:ReferenceProperties/{any}`
- 636 Each child element of `ReferenceProperties` represents an individual reference property.
- 637 `wsa:EndpointReference/wsa:ReferenceParameters/`
- 638 This optional element contains any number of individual parameters that are associated with the
639 endpoint to facilitate a particular interaction. Reference parameters are XML elements that are
640 required to properly interact with the endpoint. Reference parameters are also provided by the
641 issuer of the endpoint reference and are otherwise assumed to be opaque to consuming
642 applications.
- 643 See 5.4 for some WS-Management-specific reference parameters.
- 644 `wsa:EndpointReference/wsa:ReferenceParameters/{any}`
- 645 Each child element of `ReferenceParameters` represents an individual reference parameter.
- 646 `wsa:EndpointReference/wsa:PortType`
- 647 This optional element (of type `xs:QName`) specifies the value of the primary `portType` of the
648 endpoint being conveyed.
- 649 NOTE: The use of `wsa:PortType` is deprecated.
- 650 `wsa:EndpointReference/wsa:ServiceName`
- 651 This optional element (of type `xs:QName`) specifies the `<wsdl:service>` definition that contains a
652 WSDL description of the endpoint being referenced. The service name provides a link to a full
653 description of the service endpoint. An optional non-qualified name identifies the specific port in
654 the service that corresponds to the endpoint.
- 655 NOTE: The use of `wsa:ServiceName` is deprecated.
- 656 `wsa:EndpointReference/wsa:ServiceName/@PortName`
- 657 This optional attribute (of type `xs:NCName`) specifies the name of the `<wsdl:port>` definition that
658 corresponds to the endpoint being referenced.
- 659 `wsa:EndpointReference/wsp:Policy`
- 660 This optional element specifies a policy that is relevant to the interaction with the endpoint.
661 NOTE: The use of `wsp:Policy` is deprecated.
- 662 `wsa:EndpointReference/{any}`
- 663 This is an extensibility mechanism to allow additional elements to be specified.

664 wsa:EndpointReference/{@any}

665 This is an extensibility mechanism to allow additional attributes to be specified.

666 EXAMPLE: The following example illustrates an endpoint reference. This element references the URI
667 "http://www.fabrikam123.example/acct":

```
668 <wsa:EndpointReference xmlns:wsa="..." xmlns:fabrikam="...">
669   <wsa:Address>http://www.fabrikam123.example/acct</wsa:Address>
670 </wsa:EndpointReference>
```

671 5.1.2.2 Binding Endpoint References

672 When a message needs to be addressed to the endpoint, the information contained in the endpoint
673 reference is mapped to the message according to a transformation that is dependent on the protocol
674 and data representation used to send the message. Protocol-specific mappings (or bindings) define
675 how the information in the endpoint reference is copied to message and protocol fields. This clause
676 defines the SOAP binding for endpoint references. This mapping may be explicitly replaced by other
677 bindings (defined as WSDL bindings or as policies); however, in the absence of an applicable policy
678 stating that a different mapping is to be used, the SOAP binding defined here is assumed to apply. To
679 ensure interoperability with a broad range of devices, all conformant implementations shall support
680 the SOAP binding.

681 The SOAP binding for endpoint references is defined by the following two rules:

682 **R5.1.2.2-1:** The wsa:Address element in the endpoint reference shall be copied in the wsa:To
683 header field of the SOAP message.

684 **R5.1.2.2-2:** Each Reference Property and Reference Parameter element becomes a header
685 block in the SOAP message. The elements of each Reference Property or Reference Parameter
686 (including all of its child elements, attributes, and in-scope namespaces) shall be added as a
687 header block in the new message.

688 EXAMPLE: The following example shows how the default SOAP binding for endpoint references is used to
689 construct a message addressed to the endpoint:

```
690 <wsa:EndpointReference xmlns:wsa="..." xmlns:fabrikam="...">
691   <wsa:Address>http://www.fabrikam123.example/acct</wsa:Address>
692   <wsa:ReferenceParameters>
693     <fabrikam:CustomerKey>123456789</fabrikam:CustomerKey>
694     <fabrikam:ShoppingCart>ABCDEFGF</fabrikam:ShoppingCart>
695   </wsa:ReferenceParameters>
696 </wsa:EndpointReference>
```

697 According to the mapping rules stated before, the address value is copied in the "To" header and the
698 "CustomerKey" element should be copied literally as a header in a SOAP message addressed to this
699 endpoint. The SOAP message would look as follows:

```
700 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
701   xmlns:wsa="..." xmlns:fabrikam="..." ">
702   <S:Header>
703     ...
704     <wsa:To>http://www.fabrikam123.example/acct</wsa:To>
705     <fabrikam:CustomerKey>123456789</fabrikam:CustomerKey>
706     <fabrikam:ShoppingCart>ABCDEFGF</fabrikam:ShoppingCart>
707     ...
708   </S:Header>
709   <S:Body>
710     ...
711   </S:Body>
```

712 </S:Envelope>

713 5.1.3 Message Information Headers

714 This clause defines the syntax of a message information header.

715 The message information headers collectively augment a message with the headers shown in
716 Figure 1. These headers enable the identification and location of the endpoints involved in an
717 interaction. The basic interaction pattern from which all others are composed is "one way". In this
718 pattern a source sends a message to a destination without any further definition of the interaction.

719 "Request Reply" is a common interaction pattern that consists of an initial message sent by a source
720 endpoint (the request) and a subsequent message sent from the destination of the request back to
721 the source (the reply). A reply can be an application message, a fault, or any other message.

722 The message information header blocks provide end-to-end characteristics of a message that can be
723 easily secured as a unit. The information in these headers is immutable and not intended to be
724 modified along the message path.

725 Figure 1 shows the contents of the message information header blocks:

```
726 <wsa:MessageID> xs:anyURI </wsa:MessageID>
727 <wsa:RelatesTo RelationshipType="..."?>xs:anyURI</wsa:RelatesTo>
728 <wsa:To>xs:anyURI</wsa:To>
729 <wsa:Action>xs:anyURI</wsa:Action>
730 <wsa:From>endpoint-reference</wsa:From>
731 <wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
732 <wsa:FaultTo>endpoint-reference</wsa:FaultTo>
```

733 Figure 1 – Message Information Header Blocks

734 The following describes the attributes and elements listed in Figure 1:

735 wsa:MessageID

736 This optional element (of type xs:anyURI) uniquely identifies this message in time and space. This
737 element shall be present if wsa:ReplyTo or wsa:FaultTo is present. No two messages with a
738 distinct application intent may share a wsa:MessageID value. A message may be retransmitted for
739 any purpose (including communications failure) and may use the same wsa:MessageID value.
740 The value of this header is an opaque URI whose interpretation beyond equivalence is not defined
741 in this specification. If a reply is expected, this property shall be present.

742 wsa:RelatesTo

743 This optional (repeating) element indicates how this message relates to another message, in the
744 form of a URI-QName pair. The child of this element (which is of type xs:anyURI) contains the
745 wsa:MessageID of the related message or the following well-known URI that means "unspecified
746 message":

747 <http://schemas.xmlsoap.org/ws/2004/08/addressing/id/unspecified>

748 A reply message shall contain a wsa:RelatesTo header consisting of wsa:Reply and the
749 wsa:MessageID value of the request message.

750 wsa:RelatesTo/@RelationshipType

751 This optional attribute (of type xs:QName) conveys the relationship type as a QName. When
752 absent, the implied value of this attribute is wsa:Reply.

753 This specification has one predefined relationship type, as shown in Table 1:

754 **Table 1 – Relationship Type**

QName	Description
wsa:Reply	Indicates that this is a reply to the message identified by the URI.

755 wsa:ReplyTo

756 This optional element (of type wsa:EndpointReferenceType) provides an endpoint reference that
 757 identifies the intended receiver for replies to this message. This element shall be present if a reply
 758 is expected. If this element is present, wsa:MessageID shall be present. If a reply is expected, a
 759 message shall contain a wsa:ReplyTo header. The sender shall use the contents of the
 760 wsa:ReplyTo to formulate the reply message as defined in 5.1.3.1. If the wsa:ReplyTo header is
 761 absent, the contents of the wsa:From header may be used to formulate a message to the source.
 762 This header may be absent if the message has no meaningful reply.

763 wsa:From

764 This optional element (of type wsa:EndpointReferenceType) provides a reference to the endpoint
 765 where the message originated.

766 wsa:FaultTo

767 This optional element (of type wsa:EndpointReferenceType) provides an endpoint reference that
 768 identifies the intended receiver for faults related to this message. If this element is present,
 769 wsa:MessageID shall be present. When formulating a fault message as defined in 5.1.3.1, the
 770 sender shall use the contents of this header to formulate the fault message. If this header is
 771 absent, the sender should use the contents of the wsa:ReplyTo header to formulate the fault
 772 message. If both the wsa:FaultTo and wsa:ReplyTo header are absent, the sender may use the
 773 contents of the wsa:From header to formulate the fault message.

774 wsa:To

775 This required element (of type xs:anyURI) provides the address of the intended receiver of this
 776 message.

777 wsa:Action

778 This required element (of type xs:anyURI) uniquely identifies the semantics implied by this
 779 message. It is recommended that the value of this header be a URI identifying an input, output, or
 780 fault message within a WSDL port type. An action may be explicitly or implicitly associated with
 781 the corresponding WSDL definition. Finally, if in addition to the wsa:Action header, a SOAP Action
 782 URI is encoded in a request, the URI of the SOAP Action shall either be the same as the one
 783 specified by the wsa:Action header, or set to "".

784 The dispatching of incoming messages is based on two message properties. The mandatory wsa:To
 785 and wsa:Action header identify the target processing location and the verb or intent of the message.

786 Due to the range of network technologies currently in wide-spread use (for example, NAT, DHCP,
 787 and firewalls), many deployments cannot assign a meaningful global URI to a given endpoint. To
 788 allow these "anonymous" endpoints to initiate message exchange patterns and receive replies,
 789 Addressing defines the following well-known URI for use by endpoints that cannot have a stable,
 790 resolvable URI:

791 <http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous>

792 Requests whose wsa:ReplyTo, wsa:From and/or wsa:FaultTo headers use this address shall provide
 793 some out-of-band mechanism for delivering replies or faults (for example, returning the reply on the
 794 same transport connection). This mechanism may be a simple request/reply transport protocol (for
 795 example, HTTP GET or POST). This URI may be used as the wsa:To header for reply messages and
 796 should not be used as the wsa:To header in other circumstances.

797 5.1.3.1 Formulating a Reply Message

798 The reply to an Addressing compliant request message shall be constructed according to the rules
 799 defined in this clause.

800 EXAMPLE 1: The following example illustrates a request message using message information header blocks in a
 801 SOAP 1.2 message:

```
802 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
803   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
804   xmlns:f123="http://www.fabrikam123.example/svc53">
805   <S:Header>
806     <wsa:MessageID>uuid:aaaabbbb-cccc-dddd-eeee-fffffffffffff
807     </wsa:MessageID>
808     <wsa:ReplyTo>
809     <wsa:Address>http://business456.example/client1</wsa:Address>
810     </wsa:ReplyTo>
811     <wsa:To S:mustUnderstand="1">mailto:joe@fabrikam123.example</wsa:To>
812     <wsa:Action>http://fabrikam123.example/mail/Delete</wsa:Action>
813   </S:Header>
814   <S:Body>
815     <f123>Delete>
816       <maxCount>42</maxCount>
817     </f123>Delete>
818   </S:Body>
819 </S:Envelope>
```

820 EXAMPLE 2: The following example illustrates a reply message using message information header blocks in a
 821 SOAP 1.2 message:

```
822 <S:Envelope
823   xmlns:S="http://www.w3.org/2003/05/soap-envelope"
824   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
825   xmlns:f123="http://www.fabrikam123.example/svc53">
826   <S:Header>
827     <wsa:MessageID>
828       uuid:aaaabbbb-cccc-dddd-eeee-wwwwwwwwww
829     </wsa:MessageID>
830     <wsa:RelatesTo>
831       uuid:aaaabbbb-cccc-dddd-eeee-fffffffffffff
832     </wsa:RelatesTo>
833     <wsa:To>
834       http://business456.example/client1
835     </wsa:To>
836     <wsa:Action>http://fabrikam123.example/mail/DeleteAck</wsa:Action>
837   </S:Header>
838   <S:Body>
839     <f123>DeleteAck/>
840   </S:Body>
841 </S:Envelope>
```

842 5.1.3.2 Associating Action with WSDL Operations

843 Addressing defines two mechanisms, explicit association and default action pattern, to associate an
844 action with input, output, and fault elements within a WSDL port type.

845 5.1.3.2.1 Explicit Association

846 The action may be explicitly associated using the `wsa:Action` attribute.

847 EXAMPLE: Consider the following WSDL excerpt:

```
848 <definitions targetNamespace="http://example.com/stockquote" ...>
849   ...
850   <portType name="StockQuotePortType">
851     <operation name="GetLastTradePrice">
852       <input message="tns:GetTradePricesInput"
853         wsa:Action="http://example.com/GetQuote"/>
854       <output message="tns:GetTradePricesOutput"
855         wsa:Action="http://example.com/Quote"/>
856     </operation>
857   </portType>
858   ...
859 </definitions>
```

860 The action for the input of the `GetLastTradePrice` operation within the `StockQuotePortType` is explicitly defined to
861 be `http://example.com/GetQuote`. The action for the output of this same operation is `http://example.com/Quote`.

862 5.1.3.2.2 Default Action Pattern

863 In the absence of the `wsa:Action` attribute, the following pattern is used to construct a default action
864 for inputs and outputs. The general form of an action URI is as follows:

```
865 targetNamespace/portTypeName/(inputName|outputName)
```

866 The `/` is a literal character to be included in the action. The values of the properties are as follows:

- 867 • *targetNamespace* is the target namespace (`/definition/@targetNamespace`). If *target*
868 *namespace* ends with a `/` an additional `/` is not added.
- 869 • *portTypeName* is the name of the port type (`/definition/portType/@name`).
- 870 • *(inputName|outputName)* is the name of the element as defined in Section 2.4.5 of
871 [WSDL 1.1](#).

872 For fault messages, this pattern is not applied. Instead, the following URI is the default action URI for
873 fault messages:

```
874 http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
```

875 EXAMPLE: Consider the following WSDL excerpt:

```
876 <definitions targetNamespace="http://example.com/stockquote" ...>
877   ...
878   <portType name="StockQuotePortType">
879     <operation name="GetLastTradePrice">
880       <input message="tns:GetTradePricesInput" name="GetQuote"/>
881       <output message="tns:GetTradePricesOutput" name="Quote"/>
882     </operation>
883   </portType>
884   ...
885 </definitions>
```

886 *targetNamespace* = `http://example.com/stockquote`

887 *portTypeName* = StockQuotePortType

888 *inputName* = GetQuote

889 *outputName* = Quote

890 Applying the preceding pattern with these values produces the following:

891 input action = http://example.com/stockquote/StockQuotePortType/GetQuote

892 output action = http://example.com/stockquote/StockQuotePortType/Quote

893 WSDL defines rules for a default input or output name if the name attribute is not present. Consider
894 the following example:

895 EXAMPLE: The following is a WSDL excerpt:

```
896 <definitions targetNamespace="http://example.com/stockquote" ...>  
897   ...  
898   <portType name="StockQuotePortType">  
899     <operation name="GetLastTradePrice">  
900       <input message="tns:GetTradePricesInput"/>  
901       <output message="tns:GetTradePricesOutput"/>  
902     </operation>  
903   </portType>  
904   ...  
905 </definitions>
```

906 *targetNamespace* = http://example.com/stockquote

907 *portTypeName* = StockQuotePortType

908 According to the rules defined in 2.4.5 of [WSDL](#), if the name attribute is absent for the input of a
909 request response operation, the default value is the name of the operation with "Request" appended.

910 *inputName* = GetLastTradePriceRequest

911 Likewise, the output defaults to the operation name with "Response" appended.

912 *outputName* = GetLastTradePriceResponse

913 Applying the previous pattern with these values produces the following:

914 input action = http://example.com/stockquote/StockQuotePortType/GetLastTradePriceRequest

915 output action = http://example.com/stockquote/StockQuotePortType/GetLastTradePriceResponse

916 5.2 Versions of Addressing

917 To maintain compatibility with implementations of previous versions of WS-Management, this protocol
918 accommodates messages formatted by those previous versions. However, WS-Management 1.2 and
919 1.1 also allow for the optional use of the [WS-Addressing W3C Recommendation](#).

920 The following abbreviations are used for clarity and brevity.

- 921 • "WSMA" refers to the version of Management Addressing as specified in 5.1.
- 922 • "WSA-Rec" refers to the WS-Addressing W3C Recommendation.
- 923 • "WS-Man 1.0" refers to the *WS-Management Specification* 1.0 and implementations
924 compatible with that specification.

- 925 • "WS-Man 1.2" refers to this specification and implementations compatible with this
926 specification.
- 927 • "Addressing Anonymous URI" refers to the anonymous URI that is defined by the version of
928 Addressing currently in use. The anonymous URI defined by WSA-Rec is
929 <http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous>. The anonymous URI
930 defined by WSMA is <http://www.w3.org/2005/08/addressing/anonymous>.

931 NOTE: Some information in this clause is implementation advice to clients on algorithms for efficient
932 communication with unknown services. This informative advice should not be construed to place normative
933 requirements on the behavior of compliant clients or services.

934 **5.2.1 Technical Differences**

935 The [WSMA](#) and [WSA-Rec](#) specifications reference different XML namespaces. An endpoint sending
936 Web service messages shall use, for the Addressing SOAP headers, one namespace or the other; a
937 receiving endpoint may recognize one namespace or both namespaces. Existing implementations of
938 WS-Man 1.0 are limited to recognizing only the WSMA namespace. Interactions between WS-Man
939 1.0 and WS-Man 1.2 or 1.1 implementations will have to allow for these limitations.

940 **5.3 Requirements for Compatibility**

941 To maximize interoperability of WS-Management implementations, WS-Man 1.0 , WS-Man 1.1, and
942 WS-Man 1.2 clients and services need to be able to exchange messages. These requirements are
943 summarized in Table 2.

944 **Table 2 – Interoperability Requirements**

Interoperability Requirements between WS-Management Versions	WS-Man 1.0 Service	WS-Man 1.1 Service	WS-Man 1.2 Service
WS-Man 1.0 client	It works.	WS-Man 1.0 client needs to be able to access WS-Man 1.1 service, but some negotiation might be needed.	It works, but some negotiations might be needed.
WS-Man 1.1 client	WSMan 1.1 client needs to be able to access 1.0 service.	It works, but some negotiations might be needed.	It works, but some negotiations might be needed.
WS-Man 1.2 client	It works, but some negotiations are required.	It works, but some negotiations are required.	It works.

945 Homogeneous pairings of compliant clients and services (that is, a version 1.0 client with a version
946 1.0 service, or a version 1.2 client with a version 1.2 service) can exchange messages in accordance
947 with their respective specifications. To ensure reliable communications, heterogeneous pairings need
948 to meet certain requirements and implement certain sequencing strategies.

949 In particular, clients and services that implement WS-Man 1.0 can use only WSMA in any exchanges;
950 therefore, all exchanges with version 1.0 endpoints use only WSMA. This conclusion is summarized
951 in Table 3.

952

Table 3 – WSA Versions in Exchanges

Interoperable Version of Addressing	WS-Man 1.0 Service	WS-Man 1.1 or WS-Man 1.2 Service
WS-Man 1.0 client	WSMA	WSMA
WS-Man 1.1 or WS-Man 1.2 client	WSMA	WSMA or WSA-Rec

953 5.3.1 Discovery or Negotiation

954 If it is possible for a client to determine the capabilities of the service with respect to WSA, such
 955 discovery is more efficient than negotiating the WSA version. For instance, if a service supports
 956 Identify, then a client can determine in advance the WS-Man protocol, as well as an Addressing
 957 version or versions supported by the service. For this reason, support of Identify is mandatory in this
 958 specification when [WSA-Rec](#) is used.

959 Identify would be used as follows:

- 960 • The client sends the service an Identify message.
- 961 • If the service does not support Identify, the client can conclude that the service is a WS-
 962 Man 1.0 implementation and only supports WSMA.
- 963 • If the service successfully processes the Identify message, the client examines the versions
 964 of Addressing by looking at the AddressingVersionURI element (as defined in clause 11), if
 965 present, and can choose the appropriate version.
- 966 • If the Identify response message does not contain any Addressing versions, then there is
 967 no way for the client to know which version of Addressing to use and it would need to use
 968 one of the strategies described in 5.3.2.

969 In any case, to avoid unnecessary re-discovery or re-negotiation, a WS-Man 1.1 or 1.2 client should
 970 retain information about the capabilities of service endpoints where practical.

971 5.3.2 Client Negotiation Strategies

972 A compliant WS-Man 1.0 client will use only WSMA in message exchanges. A WS-Man 1.1 or WS-
 973 Man 1.2 client, however, may use either WSMA or WSA-Rec in message exchanges. If a WS-Man
 974 client does not know the WSA version capabilities of a service, it may use different strategies when
 975 initially contacting the service. The client may begin a message exchange with either version of WSA,
 976 using WSA-Rec or WSMA in the request message. The message exchange would proceed as
 977 follows:

- 978 • Strategy type 1: A client sends the request using WSA-Rec. The WSA-Rec SOAP headers
 979 need to be marked with a mustUnderstand="1" attribute to ensure that a fault will be
 980 generated if the receiver does not support the WSA-Rec version of Addressing. The client
 981 can then retry the operation using WSMA.
- 982 • Strategy type 2: A client sends the request using WSMA. Both WS-Man 1.0 services and
 983 WS-Man versions 1.1 and later services respond to the request using WSMA.

984 5.3.3 Initiating Message Exchanges

985 Outgoing messages initiated by a WS-Man implementation need to use the same version of
 986 Addressing that was used in the Endpoint Reference to which those messages are being sent. For
 987 example, if a Subscribe request message uses WSA-Rec in the SOAP headers (for example, for the
 988 wsa:To and wsa:ReplyTo), but uses WSMA for the NotifyTo EPR, then the Subscribe response will
 989 be sent using WSA-Rec, but the events will be sent using WSMA.

990 5.3.4 Normative Rules

991 **R5.3.4-1:** If a WS-Man service supports WSA-Rec, then it shall also support the Identify
992 operation.

993 **R5.3.4-2:** A WS-Man service version 1.1 or later shall support WSMA and should support
994 WSA-Rec.

995 **R5.3.4-3:** A WS-Man implementation that is version 1.1 or later shall send messages to
996 endpoints using the same version of Addressing used in the Endpoint Reference of the
997 destination endpoint (see 5.2).

998 **R5.3.4-4:** Within a single SOAP message, a WS-Man implementation shall use the same
999 version of Addressing for all Addressing SOAP headers.

1000 Because WS-Man version 1.1 or later allows for either version of Addressing to be used, R5.3.4-4
1001 removes the possibility of mixing the two versions for the WSA SOAP headers, but it does not
1002 disallow Endpoint References that might appear elsewhere in the message to be of a different
1003 version.

1004 In order to provide a migration path from the WSMA to WSA-Rec, the schema of certain messages
1005 allows for either version's EndpointReferenceType to be used. While the schema itself is written in a
1006 very generic way (that is, using an xs:any) allowing any arbitrary XML to appear, implementations
1007 shall restrict the contents of this element to one of the EndpointReference Types.

1008 NOTE: This allows existing WS-Man 1.0 implementations to be compliant, while providing newer
1009 implementations a migration path. In this spirit, newer implementations are strongly encouraged to support both
1010 versions of Addressing.

1011 5.4 Use of Addressing in WS-Management

1012 This clause describes the use of Endpoint References regardless of whether an implementation uses
1013 WS-Management Addressing (see 5.1) or the W3C Recommendation version of WS-Addressing.

1014 Addressing (either addressing type) endpoint references (EPRs) are used to convey information
1015 needed to address a Web service endpoint. WS-Management defines a default addressing model
1016 that can optionally be used in EPRs.

1017 5.4.1 Use of Endpoint References

1018 WS-Management uses EPRs as the addressing mechanism for individual resource instances.
1019 WS-Management also defines a default addressing model for use in addressing resources. In cases
1020 where this default addressing model is not appropriate, such as in systems with well-established
1021 addressing models or with EPRs retrieved from a discovery service, services may use those service-
1022 specific addressing models if they are based on either addressing version supported by WS-
1023 Management.

1024 **R5.4.1-1:** All messages that are addressed to a resource class or instance that is referenced
1025 by an EPR must follow the Addressing rules for representing content from the EPR (the address
1026 and reference parameters) in the SOAP message. This rule also applies to continuation
1027 messages such as Pull or Release, which continue an operation begun in a previous message.
1028 Even though such messages contain contextual information that binds them to a previous
1029 operation, the information from the EPR is still required in the message to help route it to the
1030 correct handler.

1031 Rule R5.4.1-1 clarifies that messages such as Pull or Renew still require a full EPR. For Pull, for
1032 example, this EPR would be the same as the original Enumerate, even though EnumerateResponse

1033 returns a context object that would seem to obviate the need for the EPR. The EPR is still required to
 1034 route the message properly. Similarly, the Renew request uses the SubscriptionManager EPR
 1035 received in the SubscribeResponse.

1036 When a service includes an EPR in a response message, it must be willing to accept subsequent
 1037 request messages targeted to that EPR for the same individual managed resource. Clients are not
 1038 required to process or enhance EPRs given to them by the service before using them to address a
 1039 managed resource.

1040 **R5.4.1-2:** An EPR returned by a service shall be acceptable to that service to refer to the
 1041 same managed resource.

1042 **R5.4.1-3:** All EPRs returned by a service, whether expressed using the WS-Management
 1043 default addressing model (see 5.4.2) or any other addressing model, shall be valid as long as the
 1044 managed resource exists.

1045 5.4.2 WS-Management Default Addressing Model

1046 WS-Management defines a default addressing model for resources. A service is not required to use
 1047 this addressing model, but it is suitable for many new implementations and can increase the chances
 1048 of successful interoperability between clients and services.

1049 This document uses examples of this addressing model that contain its component parts, the
 1050 ResourceURI and SelectorSet SOAP headers. This specification is independent of the actual data
 1051 model and does not define the structure of the ResourceURI or the set of values for selectors for a
 1052 given resource. These may be vendor specific or defined by other specifications.

1053 Description and use of this addressing model in this specification do not indicate that support for this
 1054 addressing model is a requirement for a conformant service.

1055 All of the normative text, examples, and conformance rules in 5.4.2 and 5.4.2.2 presume that the
 1056 service is based on the default addressing model. In cases where this addressing model is not in use,
 1057 these rules do not apply.

1058 The default addressing model uses a representation of an EPR that is a tuple of the following SOAP
 1059 headers:

- 1060 • **wsa:To** (required): the transport address of the service
- 1061 • **wsman:ResourceURI** (required if the default addressing model is used): the URI of the
 1062 resource class representation or instance representation
- 1063 • **wsman:SelectorSet** (optional): a header that identifies or "selects" the resource instance to
 1064 be accessed if more than one instance of a resource class exists

1065 The wsman:ResourceURI value needs to be marked with an s:mustUnderstand attribute set to "true"
 1066 in all messages that use the default addressing model. Otherwise, a service that does not understand
 1067 this addressing model might inadvertently return a resource that was not requested by the client.

1068 The WS-Management default addressing model is defined in the following XML outline for an EPR:

```

1069 (1) <wsa:EndpointReference>
1070 (2)   <wsa:Address>
1071 (3)     Network address
1072 (4)   </wsa:Address>
1073 (5)   <wsa:ReferenceParameters>
1074 (6)     <wsman:ResourceURI> resource URI </wsman:ResourceURI>
1075 (7)     <wsman:SelectorSet>
1076 (8)       <wsman:Selector Name="selector-name"> *
1077 (9)       Selector-value
  
```

```

1078     (10)     </wsman:Selector>
1079     (11)     </wsman:SelectorSet> ?
1080     (12)     </wsa:ReferenceParameters>
1081     (13) </wsa:EndpointReference>

```

1082 The following definitions provide additional, normative constraints on the preceding outline:

1083 **wsa:Address**

1084 the URI of the transport address

1085 **wsa:ReferenceParameters/wsman:ResourceURI**

1086 the URI of the resource class or instance to be accessed

1087 Typically, this URI represents the resource class, but it may represent the instance. The
 1088 combination of this URI and the **wsa:To** URI form the full address of the resource class or
 1089 instance.

1090 **wsa:ReferenceParameters/wsman:SelectorSet:**

1091 the optional set of selectors as described in 5.4.2.2

1092 These values are used to select an instance if the **ResourceURI** identifies a multi-instanced
 1093 target.

1094 When the default addressing model is used in a SOAP message, Addressing specifies that
 1095 translations take place and the headers are flattened out.

1096 **EXAMPLE:** The following is an example EPR definition:

```

1097     (1) <wsa:EndpointReference>
1098     (2)   <wsa:Address> Address </wsa:Address>
1099     (3)   <wsa:ReferenceParameters xmlns:wsman="...">
1100     (4)     <wsman:ResourceURI>resURI</wsman:ResourceURI>
1101     (5)     <wsman:SelectorSet>
1102     (6)       <wsman:Selector Name="Selector-name">
1103     (7)         Selector-value
1104     (8)       </wsman:Selector>
1105     (9)     </wsman:SelectorSet>
1106     (10)  </wsa:ReferenceParameters>
1107     (11) </wsa:EndpointReference>

```

1108 This address definition is translated as follows when used in a SOAP message. **wsa:Address** becomes **wsa:To**,
 1109 and the reference parameters are unwrapped and juxtaposed. The following example shows a sample SOAP
 1110 message using WSMA:

```

1111     (1) <s:Envelope xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1112     (2)   <s:Header>
1113     (3)     <wsa:To> Address </wsa:To>
1114     (4)     <wsa:Action> Action URI </wsa:Action>
1115     (5)     <wsman:ResourceURI s:mustUnderstand="true">resURI</wsman:ResourceURI>
1116     (6)     <wsman:SelectorSet>
1117     (7)       <wsman:Selector Name="Selector-name">
1118     (8)         Selector-value
1119     (9)       </wsman:Selector>
1120     (10)    </wsman:SelectorSet>
1121     (11)    ...
1122     (12)   </s:Header>
1123     (13)   <s:Body> ... </s:Body>
1124     (14) </s:Envelope>

```

1125 The following message shows a sample SOAP message using WS-Rec:

```

1126     (1) <s:Envelope xmlns:wsa="http://www.w3.org/2005/08/addressing ">

```

```

1127 (2) <s:Header>
1128 (3)   <wsa:To s:mustUnderstand="true"> Address </wsa:To>
1129 (4)   <wsa:Action s:mustUnderstand="true"> Action URI </wsa:Action>
1130 (5)   <wsman:ResourceURI s:mustUnderstand="true"
1131 (6)     wsa:isReferenceParameter="true">resURI</wsman:ResourceURI>
1132 (7)   <wsman:SelectorSet wsa:isReferenceParameter="true">
1133 (8)     <wsman:Selector Name="Selector-name">
1134 (9)       Selector-value
1135 (10)    </wsman:Selector>
1136 (11)  </wsman:SelectorSet>
1137 (12)  ...
1138 (13) </s:Header>
1139 (14) <s:Body> ... </s:Body>
1140 (15) </s:Envelope>

```

1141 In both cases, the `wsa:To`, `wsman:ResourceURI`, and `wsman:SelectorSet` elements work together to
 1142 *reference* the resource instance to be managed, but the actual *method* or *operation* to be executed
 1143 against this resource is indicated by the `wsa:Action` header.

1144 **EXAMPLE:** The following is an example of Addressing headers based on the default addressing model in an
 1145 actual message:

```

1146 (1) <s:Envelope
1147 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1148 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1149 (4)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
1150 (5)   <s:Header>
1151 (6)     ...
1152 (7)     <wsa:To>http://123.99.222.36/wsman</wsa:To>
1153 (8)     <wsman:ResourceURI s:mustUnderstand="true">
1154 (9)       http://example.org/hardware/2005/02/storage/physDisk
1155 (10)    </wsman:ResourceURI>
1156 (11)    <wsman:SelectorSet>
1157 (12)      <wsman:Selector Name="LUN"> 2 </wsman:Selector>
1158 (13)    </wsman:SelectorSet>
1159 (14)    <wsa:Action> http://schemas.xmlsoap.org/ws/2004/09/transfer/Get
1160 (15)    </wsa:Action>
1161 (16)    <wsa:MessageID> urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a91
1162 (17)    </wsa:MessageID>
1163 (18)    ...
1164 (19)  </s:Header>
1165 (20) <s:Body> ... </s:Body>
1166 (21) </s:Envelope>

```

1167 The following definitions apply to the preceding message example:

1168 **wsa:To**

1169 the network (or transport-level) address of the service

1170 **wsman:ResourceURI**

1171 the ResourceURI of the resource class or resource instance to be accessed

1172 **wsman:SelectorSet**

1173 a wrapper for the selectors

1174 **wsman:SelectorSet/wsman:Selector**

1175 identifies or selects the resource instance to be accessed, if more than one instance of the
 1176 resource exists

1177 In this case, the selector is "LUN" (logical unit number), and the selected device is unit number
1178 "2".

1179 wsa:Action

1180 identifies which operation is to be carried out against the resource (in this case, a "Get")

1181 wsa:MessageID

1182 identifies this specific message uniquely for tracking and correlation purposes

1183 The format defined in [RFC 4122](#) is often used in the examples in this specification, but it is not
1184 required.

1185 5.4.2.1 ResourceURI

1186 The ResourceURI is used to indicate the class resource or instance.

1187 **R5.4.2.1-1:** The format of the wsman:ResourceURI is unconstrained provided that it meets [RFC](#)
1188 [3986](#) requirements.

1189 The format and syntax of the ResourceURI is any valid URI according to [RFC 3986](#). Although there is
1190 no default scheme, http: and urn: are common defaults. If http: is used, users may expect to find
1191 Web-based documentation of the resource at that address. The wsa:To and the wsman:ResourceURI
1192 elements work together to define the actual resource being targeted.

1193 **R5.4.2.1-2:** Vendor-specific or organization-specific URIs should contain the Internet domain
1194 name in the first token sequence after the scheme, such as "example.org" in ResourceURI in the
1195 following example.

1196 EXAMPLE:

```
1197 (20) <s:Header>  
1198 (21)   <wsa:To> http://123.15.166.67/wsman </wsa:To>  
1199 (22)   <wsman:ResourceURI>  
1200 (23)     http://schemas.example.org/2005/02/hardware/physDisk  
1201 (24)   </wsman:ResourceURI>  
1202 (25)   ...  
1203 (26) </s:Header>
```

1204 **R5.4.2.1-3:** When the default addressing model is used, the wsman:ResourceURI reference
1205 parameter is required in messages with the following wsa:Action URIs:

1206 http://schemas.xmlsoap.org/ws/2004/09/transfer/Get
1207 http://schemas.xmlsoap.org/ws/2004/09/transfer/Put
1208 http://schemas.xmlsoap.org/ws/2004/09/transfer/Create
1209 http://schemas.xmlsoap.org/ws/2004/09/transfer/Delete
1210 http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate
1211 http://schemas.xmlsoap.org/ws/2004/09/enumeration/Pull
1212 http://schemas.xmlsoap.org/ws/2004/09/enumeration/Renew
1213 http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatus
1214 http://schemas.xmlsoap.org/ws/2004/09/enumeration/Release
1215 http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe

- 1216 The following messages require the EPR to be returned in the SubscriptionManager element of the
1217 SubscribeResponse message. The format of the EPR is determined by the service and might or
1218 might not include the ResourceURI:
- 1219 `http://schemas.xmlsoap.org/ws/2004/08/eventing/Renew`
1220 `http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatus`
- 1221 While the ResourceURI SOAP header is required when the WS-Management default addressing
1222 mode is used, it may be short and of a very simple form, such as `http://example.com/*` or
1223 `http://example.com/resource`.
- 1224 **R5.4.2.1-4:** For the request message of custom actions (methods), the ResourceURI header may
1225 be present in the message to help route the message to the correct handler.
- 1226 **R5.4.2.1-5:** The ResourceURI element should not appear in other messages, such as responses
1227 or events, unless the associated EPR includes it in its ReferenceParameters.
- 1228 In practice, the `wsman:ResourceURI` element is required only in requests to reference the targeted
1229 resource class. Responses are not addressed to a management resource, so the
1230 `wsman:ResourceURI` has no meaning in that context.
- 1231 **R5.4.2.1-6:** When the default addressing model is used and the `wsman:ResourceURI` element is
1232 missing or in an incorrect form, the service shall issue a `wsa:DestinationUnreachable` fault with a
1233 detail code of
- 1234 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidResourceURI`
- 1235 **R5.4.2.1-7:** The `wsman:ResourceURI` element shall be used to indicate only the identity of a
1236 resource, and it may not be used to indicate the action being applied to that resource, which is
1237 properly expressed using the `wsa:Action` URI.
- 1238 Custom WSDL-based methods have both a ResourceURI identity from the perspective of addressing
1239 and a `wsa:Action` URI from the perspective of execution. In many cases, the ResourceURI is simply a
1240 pseudonym for the WSDL identity and Port, and the `wsa:Action` URI is the specific method within that
1241 port (or interface) definition.
- 1242 Although a single URI could theoretically be used alone to define an instance of a multi-instance
1243 resource, it is recommended that the `wsa:To` element be used to locate the WS-Management service,
1244 that the `wsman:ResourceURI` element be used to identify the resource class, and that the
1245 `wsman:SelectorSet` element be used to reference the resource instance. If the resource consists of
1246 only a single instance, then the `wsman:ResourceURI` element alone refers to the single instance.
- 1247 This usage is not a strict requirement, just a guideline. The service can use distinct selectors for any
1248 given operation, even against the same resource class, and may allow or require selectors for the
1249 Enumerate operation.
- 1250 See the recommendations in 7.2 regarding addressing uniformity.
- 1251 Custom actions have two distinct identities: the ResourceURI, which can identify the WSDL and port
1252 (or interface), and the `wsa:Action` URI, which identifies the specific method. If only one method exists
1253 in the interface, in a sense the ResourceURI and `wsa:Action` URI are identical.
- 1254 It is not an error to use the `wsa:Action` URI for the ResourceURI of a custom method, but both are still
1255 required in the message for uniform processing on both clients and servers.

1256 EXAMPLE 1: The following action to reset a network card might have the following EPR usage:

```

1257 (1) <s:Header>
1258 (2)   <wsa:To>
1259 (3)     http://1.2.3.4/wsman/
1260 (4)   </wsa:To>
1261 (5)   <wsman:ResourceURI>http://example.org/2005/02/networkcards/reset
1262 (6)   </wsman:ResourceURI>
1263 (7)   <wsa:Action>
1264 (8)     http://example.org/2005/02/networkcards/reset
1265 (9)   </wsa:Action>
1266 (10)  ...
1267 (11) </s:Header>

```

1268 In many cases, the ResourceURI is equivalent to a WSDL name and port, and the wsa:Action URI
1269 contains an additional token as a suffix, as in the following example.

1270 EXAMPLE 2:

```

1271 (1) <s:Header>
1272 (2)   <wsa:To>
1273 (3)     http://1.2.3.4/wsman
1274 (4)   </wsa:To>
1275 (5)   <wsman:ResourceURI>http://example.org/2005/02/networkcards
1276 (6)   </wsman:ResourceURI>
1277 (7)   <wsa:Action>
1278 (8)     http://example.org/2005/02/networkcards/reset
1279 (9)   </wsa:Action>
1280 (10)  ...
1281 (11) </s:Header>

```

1282 Finally, the ResourceURI may be completely unrelated to the wsa:Action URI, as in the following
1283 example.

1284 EXAMPLE 3:

```

1285 (1) <s:Header>
1286 (2)   <wsa:To>http://1.2.3.4/wsman</wsa:To>
1287 (3)   <wsman:ResourceURI>
1288 (4)     http://example.org/products/management/networkcards
1289 (5)   </wsman:ResourceURI>
1290 (6)   <wsa:Action>
1291 (7)     http://example.org/2005/02/netcards/reset
1292 (8)   </wsa:Action>
1293 (9)   ...
1294 (10) </s:Header>

```

1295 All of these uses are legal.

1296 When used with subscriptions, the EPR described by wsa:Address and wsman:ResourceURI (and
1297 optionally the wsman:SelectorSet values) identifies the event source to which the subscription is
1298 directed. In many cases, the ResourceURI identifies a real or virtual event log, and the subscription is
1299 intended to provide real-time notifications of any new entries added to the log. In many cases, the
1300 wsman:SelectorSet element might not be used as part of the EPR.

1301 5.4.2.2 Selectors

1302 In the WS-Management default addressing model, selectors are optional elements used to identify
1303 instances within a resource class. For operations such as Get or Put, the selectors are used to
1304 identify a single instance of the resource class referenced by the ResourceURI.

1305 In practice, because the ResourceURI often acts as a table or a "class," the SelectorSet element is a
1306 discriminant used to identify a specific "row" or "instance." If only one instance of a resource class is
1307 implied by the ResourceURI, the SelectorSet can be omitted because the ResourceURI is acting as
1308 the full identity of the resource. If more than one selector value is required, the entire set of selectors
1309 is interpreted by the service in order to reference the specific instance. The selectors are interpreted
1310 as being separated by implied logical AND operators.

1311 In some information domains, the values referenced by the selectors are "keys" that are part of the
1312 resource content itself, whereas in other domains the selectors are part of a logical or physical
1313 directory system or search space. In these cases, the selectors are used to identify the resource, but
1314 are not part of the representation.

1315 **R5.4.2.2-1:** If a resource has more than one instance, a wsman:SelectorSet element may be
1316 used to distinguish which instance is targeted if the WS-Management default addressing model is
1317 in use. Any number of wsman:Selector values may appear with the wsman:SelectorSet element,
1318 as required to identify the precise instance of the resource class. The service may consider the
1319 case of selector names and values (see 13.6), as required by the underlying execution
1320 environment.

1321 If the client needs to discover the policy on how the case of selector values is interpreted, the service
1322 can provide metadata documents that describe this policy. The format of such metadata is beyond
1323 the scope of this specification.

1324 **R5.4.2.2-2:** All content within the SelectorSet element is to be treated as a single reference
1325 parameter with a scope relative to the ResourceURI.

1326 **R5.4.2.2-3:** A service using the WS-Management default addressing model shall examine all
1327 selectors in the message and process them as if they were logically joined by AND. If the set of
1328 selectors is incorrect for the targeted resource instance, a wsman:InvalidSelectors fault should be
1329 returned to the client with the following detail codes:

- 1330 • if selectors are missing:
1331 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsufficientSelectors>
- 1332 • if selector values are the wrong types:
1333 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/TypeMismatch>
- 1334 • if the selector value is of the correct type from the standpoint of XML types, but out of range
1335 or otherwise illegal in the specific information domain:
1336 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValue>
- 1337 • if the name is not a recognized selector name
1338 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnexpectedSelectors>

1339 **R5.4.2.2-4:** The Selector Name attribute shall not be duplicated at the same level of nesting. If
1340 this occurs, the service should return a wsman:InvalidSelectors fault with the following detail
1341 code:

1342 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DuplicateSelectors>

1343 This specification does not mandate the use of selectors. Some implementations may decide to use
1344 complex URI schemes in which the ResourceURI itself implicitly identifies the instance.

1345 The format of the SelectorSet element is as follows:

```

1346 (1) <s:Envelope
1347 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1348 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1349 (4)   xmlns:wsmn="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
1350 (5)   <s:Header>
1351 (6)     ...
1352 (7)     <wsa:To> service transport address </wsa:To>
1353 (8)     <wsman:ResourceURI> ResourceURI </wsman:ResourceURI>
1354 (9)     <wsman:SelectorSet>
1355 (10)      <wsman:Selector Name="name"> value </wsman:Selector> +
1356 (11)    </wsman:SelectorSet> ?
1357 (12)    ...
1358 (13)   </s:Header>
1359 (14)   <s:Body> ... </s:Body>
1360 (15) </s:Envelope>

```

1361 The following definitions provide additional, normative constraints on the preceding outline:

1362 **wsman:SelectorSet**

1363 the wrapper for one or more Selector elements required to reference the instance

1364 **wsman:SelectorSet/wsman:Selector**

1365 used to describe the selector and its value

1366 If more than one selector is required, one Selector element exists for each part of the overall
 1367 selector. The value of this element is the Selector value.

1368 **wsman:SelectorSet/wsman:Selector/@Name**

1369 the name of the selector (to be treated in a case-insensitive manner)

1370 The value of a selector may be a nested EPR.

1371 **EXAMPLE:** In the following example, the selector on line 9 is a part of a SelectorSet that contains a nested
 1372 EPR (lines 10–18) with its own Address, ResourceURI, and SelectorSet elements:

```

1373 (1) <s:Envelope
1374 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1375 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1376 (4)   xmlns:wsmn="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
1377 (5)   <s:Header>
1378 (6)     ...
1379 (7)     <wsman:SelectorSet>
1380 (8)       <wsman:Selector Name="Primary"> 123 </wsman:Selector>
1381 (9)       <wsman:Selector Name="EPR">
1382 (10)        <wsa:EndpointReference>
1383 (11)          <wsa:Address> address </wsa:Address>
1384 (12)          <wsa:ReferenceParameters>
1385 (13)            <wsman:ResourceURI> resource URI </wsman:ResourceURI>
1386 (14)            <wsman:SelectorSet>
1387 (15)              <wsman:Selector Name="name"> value </wsman:Selector>
1388 (16)            </wsman:SelectorSet>
1389 (17)          </wsa:ReferenceParameters>
1390 (18)        </wsa:EndpointReference>
1391 (19)      </wsman:Selector>
1392 (20)    </wsman:SelectorSet>
1393 (21)    ...
1394 (22)   </s:Header>
1395 (23)   <s:Body> ... </s:Body>
1396 (24) </s:Envelope>

```

1397 **R5.4.2.2-5:** For those services using the WS-Management default addressing model, the value of
1398 a wsman:Selector shall be one of the following values:

- 1399 • a simple type as defined in the XML schema namespace
1400 http://www.w3.org/2001/XMLSchema
- 1401 • a nested wsa:EndpointReference using the WS-Management default addressing model

1402 A service may fault selector usage with wsman:InvalidSelectors if the selector is not a simple type or
1403 an EPR.

1404 **R5.4.2.2-6:** A conformant service may reject any selector or nested selector with a nested EPR
1405 whose wsa:Address value is not the same as the primary wsa:To value or is not the Addressing
1406 Anonymous URI.

1407 The primary purpose for this nesting mechanism is to allow resources that can answer questions
1408 about other resources.

1409 **R5.4.2.2-7:** A service may fail to process a selector name of more than 2048 characters.

1410 **R5.4.2.2-8:** A service may fail to process a selector value of more than 4096 characters,
1411 including any embedded selectors, and may fail to process a message that contains more than
1412 8096 characters of content in the root SelectorSet element.

1413 **5.4.2.3 Faults for Default Addressing Model**

1414 When faults related to the information in the addressing model based on the default format are
1415 generated, they may contain specific fault detail codes. These detail codes are called out separately
1416 in 14.6 and do not apply when service-specific addressing is used.

1417 **5.4.3 Service-Specific Endpoint References**

1418 Although WS-Management specifies a default addressing model, in some cases this model is not
1419 available or appropriate.

1420 **R5.4.3-1:** A conformant service may not understand the header values used by the
1421 WS-Management default addressing model. If this is the case, and if the client marks the
1422 wsman:ResourceURI with mustUnderstand="true", the service shall return an s:NotUnderstood
1423 fault.

1424 **R5.4.3-2:** A conformant service may require additional header values to be present that are
1425 beyond the scope of this specification.

1426 Services can thus use alternative addressing models for referencing resources with
1427 WS-Management. These addressing models might or might not use ResourceURI or SelectorSet
1428 elements and still be valid addressing models if they conform to the rules of Addressing.

1429 In addition to a defined alternative addressing model, a service might not explicitly define any
1430 addressing model at all and instead use an opaque EPR generated at run-time, which is handled
1431 according to the standard rules of Addressing.

1432 When such addressing models are used, the client application has to understand and interoperate
1433 with discovery methods for acquiring EPRs that are beyond the scope of this specification.

1434 5.4.4 mustUnderstand

1435 This clause describes the use of the mustUnderstand attribute, regardless of whether an
1436 implementation uses WS-Management Addressing (see 5.1) or the W3C Recommendation type of
1437 WS-Addressing.

1438 The mustUnderstand attribute for SOAP headers is to be interpreted as a "must comply" instruction in
1439 WS-Management. For example, if a SOAP header that is listed as being optional in this specification
1440 is tagged with mustUnderstand="true", the service is required to comply or return a fault. To ensure
1441 that the service treats a header as optional, the mustUnderstand attribute can be omitted.

1442 If the wsa:Action URI is not understood, the implementation might not know how to process the
1443 message. So, for the following elements, the omission or inclusion of mustUnderstand="true" has no
1444 real effect on the message in practice, because mustUnderstand is implied:

- 1445 • wsa:To
- 1446 • wsa:MessageID
- 1447 • wsa:RelatesTo
- 1448 • wsa:Action
- 1449 • wsa:ReplyTo
- 1450 • wsa:FaultTo

1451 **R5.4.4-1:** A conformant service shall process any of the preceding elements identically
1452 regardless of whether mustUnderstand="true" is present.

1453 As a corollary, clients can omit mustUnderstand="true" from any of the preceding elements with no
1454 change in meaning.

1455 **R5.4.4-2:** If a service cannot comply with a header marked with mustUnderstand="true", it
1456 shall issue an s:NotUnderstood fault.

1457 The goal is for the service to be tolerant of inconsistent mustUnderstand usage by clients when the
1458 request is not likely to be misinterpreted.

1459 It is important that clients using the WS-Management default addressing model (ResourceURI and
1460 SelectorSet) use mustUnderstand="true" on the wsman:ResourceURI element to ensure that the
1461 service is compliant with that addressing model. Implementations that use service-specific addressing
1462 models will otherwise potentially ignore these header values and behave inconsistently with the
1463 intentions of the client.

1464 5.4.5 wsa:To

1465 This clause describes the use of the Addressing wsa:To header regardless of whether an
1466 implementation uses WS-Management Addressing (see 5.1) or the W3C Recommendation version of
1467 WS-Addressing.

1468 In request messages, the wsa:To address contains the transport address of the service. In some
1469 cases, this address is sufficient to locate the resource. In other cases, the service is a dispatching
1470 agent for multiple resources. In these cases, the message typically contains additional headers to
1471 allow the service to identify a resource within its scope. For example, when the default addressing
1472 model is in use, these additional headers will be the ResourceURI and SelectorSet elements.

1473 NOTE: WS-Management does not preclude multiple listener services from coexisting on the same physical
1474 system. Such services would be discovered and distinguished using mechanisms beyond the scope of this
1475 specification.

1476 **R5.4.5-1:** The `wsa:To` header shall be present in all messages, whether requests, responses,
 1477 or events. In the absence of other requirements, it is recommended that the network address for
 1478 resources that require authentication be suffixed by the token sequence `/wsman`. If `/wsman` is
 1479 used, unauthenticated access should not be allowed.

1480 (1) `<wsa:To> http://123.15.166.67/wsman </wsa:To>`

1481 **R5.4.5-2:** In the absence of other requirements, it is recommended that the network address
 1482 for resources that do not require authentication be suffixed by the token sequence `/wsman-anon`.
 1483 If `/wsman-anon` is used, authenticated access shall not be required.

1484 (1) `<wsa:To> http://123.15.166.67/wsman-anon </wsa:To>`

1485 Including the network transport address in the SOAP message may seem redundant because the
 1486 network connection would already be established by the client. However, in cases where the
 1487 message is routed through intermediaries, the network transport address is required so that the
 1488 intermediaries can examine the message and make the connection to the actual endpoint.

1489 The `wsa:To` header may encompass any number of tokens required to locate the service and a group
 1490 of resources within that service.

1491 **R5.4.5-3:** The service should generate a fault when the `wsa:To` address cannot be processed
 1492 due to the following situations::

1493 • If the resource is offline, a `wsa:EndpointUnavailable` fault is returned with the following
 1494 detail code:

1495 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ResourceOffline`

1496 • If the resource cannot be located ("not found"), a `wsa:DestinationUnreachable` fault is
 1497 returned.

1498 • If the resource is valid, but internal errors occur, a `wsman:InternalError` fault is returned.

1499 • If the resource cannot be accessed for security reasons, a `wsman:AccessDenied` fault is
 1500 returned.

1501 5.4.6 Other Addressing Headers

1502 This clause describes the use of other Addressing headers, regardless of whether an implementation
 1503 uses WS-Management Addressing (see 5.1) or the W3C Recommendation version of WS-
 1504 Addressing.

1505 WS-Management depends on Addressing to describe the rules for use of other Addressing headers.

1506 5.4.6.1 Processing Addressing Headers

1507 The following additional addressing-related header blocks occur in WS-Management messages.

1508 **R5.4.6.1-1:** A conformant service shall recognize and process the following Addressing header
 1509 blocks.

- 1510 • **`wsa:To`**
- 1511 • **`wsa:ReplyTo`** (required when a response is expected)
- 1512 • **`wsa:FaultTo`** (optional)
- 1513 • **`wsa:MessageID`** (required)
- 1514 • **`wsa:Action`** (required)
- 1515 • **`wsa:RelatesTo`** (required in responses)

1516 The use of these header blocks is discussed in subsequent clauses.

1517 5.4.6.2 wsa:ReplyTo

1518 WS-Management requires the following usage of wsa:ReplyTo in addressing:

1519 **R5.4.6.2-1:** A wsa:ReplyTo header shall be present in all request messages when a reply is
 1520 required. This address shall be either a valid address for a new connection using any transport
 1521 supported by the service or the Addressing Anonymous URI, which indicates that the reply is to
 1522 be delivered over the same connection on which the request arrived. If the wsa:ReplyTo header
 1523 is missing, a wsa:MessageInformationHeaderRequired fault is returned.

1524 Some messages, such as event deliveries, SubscriptionEnd, and so on, do not require a response
 1525 and may omit a wsa:ReplyTo element.

1526 **R5.4.6.2-2:** A conformant service may require that all responses be delivered over the same
 1527 connection on which the request arrives. In this case, the URI discussed in R5.4.6.2-1 shall
 1528 indicate this. Otherwise, the service shall return a wsman:UnsupportedFeature fault with the
 1529 following detail code:

1530 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode`

1531 **R5.4.6.2-3:** When delivering events for which acknowledgement of delivery is required, the
 1532 sender of the event shall include a wsa:ReplyTo element and observe the usage in 10.8 of this
 1533 specification.

1534 **R5.4.6.2-4:** This rule intentionally left blank.

1535 **R5.4.6.2-5:** This rule intentionally left blank.

1536 Addressing allows clients to include client-defined reference parameters in wsa:ReplyTo headers.
 1537 Addressing requires that these reference parameters be extracted from requests and placed in the
 1538 responses by removing the ReferenceParameters wrapper and placing all of the values as top-level
 1539 SOAP headers in the response, as discussed in 5.1. This allows clients to better correlate responses
 1540 with the original requests. This step cannot be omitted.

1541 **EXAMPLE:** In the following example, the header x:someHeader is included in the reply message:

```

1542 (1) <s:Envelope
1543 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1544 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1545 (4)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
1546 (5)   <s:Header>
1547 (6)     ...
1548 (7)     <wsa:To> http://1.2.3.4/wsman </wsa:To>
1549 (8)     <wsa:ReplyTo>
1550 (9)       <wsa:Address>
1551 (10)        http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
1552 (11)       </wsa:Address>
1553 (12)       <wsa:ReferenceParameters>
1554 (13)         <x:someHeader xmlns:x="..."> user-defined content </x:someHeader>
1555 (14)       </wsa:ReferenceParameters>
1556 (15)     </wsa:ReplyTo>
1557 (16)     ...
1558 (17)   </s:Header>
1559 (18)   <s:Body> ... </s:Body>
1560 (19) </s:Envelope>

```

1561 **R5.4.6.2-6:** If the wsa:ReplyTo address is not usable or is missing, the service should not reply to
 1562 the request and it should close or terminate the connection according to the rules of the current
 1563 network transport. In these cases, the service should locally log some type of entry to help locate
 1564 the client defect later.

1565 5.4.6.3 wsa:FaultTo

1566 WS-Management qualifies the use of wsa:FaultTo as indicated in this clause.

1567 **R5.4.6.3-1:** A conformant service may support a wsa:FaultTo address that is distinct from the
 1568 wsa:ReplyTo address. If such a request is made and is not supported by the service, a
 1569 wsman:UnsupportedFeature fault shall be returned with the following detail code:

1570 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode`

1571 If both the wsa:FaultTo and wsa:ReplyTo headers are omitted from a request, transport-level
 1572 mechanisms are typically used to fail the request because the address to which the fault is to be sent
 1573 is uncertain. In such a case, it is not an error for the service to simply shut down the connection.

1574 **R5.4.6.3-2:** If wsa:FaultTo is omitted, the service shall return the fault to the wsa:ReplyTo
 1575 address if a fault occurs.

1576 **R5.4.6.3-3:** A conformant service may require that all faults be delivered to the client over the
 1577 same transport or connection on which the request arrives. In this case, the URI shall be the
 1578 Addressing Anonymous URI. If services do not support separately addressed fault delivery and
 1579 the wsa:FaultTo is any other address, a wsman:UnsupportedFeature fault shall be returned with
 1580 the following detail code:

1581 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode`

1582 NOTE: This specification does not restrict richer implementations from fully supporting wsa:FaultTo.

1583 **R5.4.6.3-4:** This rule intentionally left blank.

1584 EXAMPLE: In the following example, the header x:someHeader is included in fault messages if they occur:

```

1585 (1) <s:Envelope
1586 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1587 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1588 (4)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
1589 (5)   <s:Header>
1590 (6)     ...
1591 (7)     <wsa:To> http://1.2.3.4/wsman </wsa:To>
1592 (8)     <wsa:FaultTo>
1593 (9)       <wsa:Address>
1594 (10)        http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
1595 (11)       </wsa:Address>
1596 (12)       <wsa:ReferenceParameters>
1597 (13)         <x:someHeader xmlns:x="..."> user-defined content </x:someHeader>
1598 (14)       </wsa:ReferenceParameters>
1599 (15)     </wsa:FaultTo>
1600 (16)     ...
1601 (17)   </s:Header>
1602 (18)   <s:Body> ... </s:Body>
1603 (19) </s:Envelope>

```

1604 **R5.4.6.3-5:** If the wsa:FaultTo address is not usable, the service should not reply to the request.
 1605 Similarly, if according to WS-Addressing processing rules there is no suitable address to send a

1606 fault to, it should not reply and should close the network connection. In these cases, the service
1607 should locally log some type of entry to help locate the client defect later.

1608 **R5.4.6.3-6:** The service shall properly duplicate the wsa:Address of the wsa:FaultTo element in
1609 the wsa:To of the reply, even if some of the information is not understood by the service.

1610 This rule applies in cases where the client includes private content suffixes on the HTTP or HTTPS
1611 address that the service does not understand. If the service removes this information when
1612 constructing the address, the subsequent message might not be correctly processed.

1613 **5.4.6.4 wsa:MessageID and wsa:RelatesTo**

1614 WS-Management qualifies the use of wsa:MessageID and wsa:RelatesTo as follows:

1615 **R5.4.6.4-1:** The MessageID and RelatesTo URIs may be of any format, as long as they are valid
1616 URIs according to [RFC 3986](#). Two URIs are considered different even if the characters in the
1617 URIs differ only by case.

1618 The following two formats are endorsed by this specification. The first is considered a best
1619 practice because it is backed by [RFC 4122](#):

1620 urn:uuid:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

1621 or

1622 uuid:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

1623 In these formats, each x is an uppercase or lowercase hexadecimal digit (lowercase is required
1624 by [RFC 4122](#)); there are no spaces or other tokens. The value may be a DCE-style universally
1625 unique identifier (UUID) with provable uniqueness properties in this format, however, it is not
1626 necessary to have provable uniqueness properties in the URIs used in the wsa:MessageID and
1627 wsa:RelatesTo headers.

1628 Regardless of format, the URI should not exceed the maximum defined in R13.1-6.

1629 UUIDs have a numeric meaning as well as a string meaning, and this can lead to confusion. A UUID
1630 in lowercase is a different URI from the same UUID in uppercase. This is because URIs are case-
1631 sensitive. If a UUID is converted to its decimal equivalent the case of the original characters is lost.
1632 WS-Management works with the URI value itself, not the underlying decimal equivalent
1633 representation. Services are free to *interpret* the URI in any way, but are not allowed to alter the case
1634 usage when repeating the message or any of the MessageID values in subsequent messages.

1635 The [RFC 4122](#) requires the digits to be lowercase, which is the responsibility of the client. The service
1636 simply processes the values as URI values and is not required to analyze the URI for correctness or
1637 compliance. The service replicates the client usage in the wsa:RelatesTo reply header and is not
1638 allowed to alter the case usage.

1639 **R5.4.6.4-2:** The MessageID should be generated according to any algorithm that ensures that no
1640 two MessageIDs are repeated. Because the value is treated as case-sensitive (**R5.4.6.4-1**),
1641 confusion can arise if the same value is reused differing only in case. As a result, the service shall
1642 not create or employ MessageID values that differ only in case. For any message transmitted by
1643 the service, the MessageID shall not be reused.

1644 The client ensures that MessageID values are not reused in requests. Although services and clients
1645 can issue different MessageIDs that differ only in case, the service is not required to detect this
1646 difference, nor is it required to analyze the URI for syntactic correctness or repeated use.

1647 **R5.4.6.4-3:** The RelatesTo element shall be present in all response messages and faults, shall
 1648 contain the MessageID of the associated request message, and shall match the original in case,
 1649 being treated as a URI value and not as a binary UUID value.

1650 **R5.4.6.4-4:** If the MessageID is not parsable or is missing, a
 1651 wsa:InvalidMessageInformationHeader fault should be returned.

1652 EXAMPLE: The following examples show wsa:MessageID usage:

```
1653 (20) <wsa:MessageID>
1654 (21)   uuid:d9726315-bc91-430b-9ed8-ce5ffb858a91
1655 (22) </wsa:MessageID>
1656 (23)
1657 (24) <wsa:MessageID>
1658 (25)   anotherScheme:ID/12310/1231/16607/25
1659 (26) </wsa:MessageID>
```

1660 5.4.6.5 wsa:Action

1661 The wsa:Action URI indicates the "operation" being invoked against the resource.

1662 **R5.4.6.5-1:** The wsa:Action URI shall not be used to identify the specific resource class or
 1663 instance, but only to identify the operation to use against that resource.

1664 **R5.4.6.5-2:** For all resource endpoints, a service shall return a wsa:ActionNotSupported fault if a
 1665 requested action is not supported by the service for the specified resource.

1666 In other words, to model the "Get" of item "Disk", the wsa:Action URI contains the "Get". The wsa:To,
 1667 and potentially other SOAP headers, indicate *what* is being accessed. When the default addressing
 1668 model is used, for example, the ResourceURI typically contains the reference to the "Disk" and the
 1669 SelectorSet identifies which disk. Other service-specific addressing models can factor the identity of
 1670 the resource in different ways.

1671 Implementations are free to support additional custom methods that combine the notion of "Get" and
 1672 "Disk" into a single "GetDisk" action if they strive to support the separated form to maximize
 1673 interoperability. One of the main points behind WS-Management is to unify common methods
 1674 wherever possible.

1675 **R5.4.6.5-3:** If a service exposes any of the following types of capabilities, a conformant service
 1676 shall at least expose that capability using the definitions in Table 4 according to the rules of this
 1677 specification. The service may optionally expose additional similar functionality using a distinct
 1678 wsa:Action URI.

1679

Table 4 – wsa:Action URI Descriptions

Action URI	Description
http://schemas.xmlsoap.org/ws/2004/09/transfer/Get	Models any simple single item retrieval
http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse	Response to "Get"
http://schemas.xmlsoap.org/ws/2004/09/transfer/Put	Models an update of an entire item
http://schemas.xmlsoap.org/ws/2004/09/transfer/PutResponse	Response to "Put"
http://schemas.xmlsoap.org/ws/2004/09/transfer/Create	Models creation of a new item
http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse	Response to "Create"
http://schemas.xmlsoap.org/ws/2004/09/transfer/Delete	Models the deletion of an item
http://schemas.xmlsoap.org/ws/2004/09/transfer/DeleteResponse	Response to "Delete"

Action URI	Description
http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate	Begins an enumeration or query
http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerateResponse	Response to "Enumerate"
http://schemas.xmlsoap.org/ws/2004/09/enumeration/Pull	Retrieves the next batch of results from enumeration
http://schemas.xmlsoap.org/ws/2004/09/enumeration/PullResponse	Response to "Pull"
http://schemas.xmlsoap.org/ws/2004/09/enumeration/Renew	Renews an enumerator that may have timed out (not required in WS-Management)
http://schemas.xmlsoap.org/ws/2004/09/enumeration/RenewResponse	Response to "Renew" (not required in WS-Management)
http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatus	Gets the status of the enumerator (not required in WS-Management)
http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatusResponse	Response to "GetStatus" (not required in WS-Management)
http://schemas.xmlsoap.org/ws/2004/09/enumeration/Release	Releases an active enumerator
http://schemas.xmlsoap.org/ws/2004/09/enumeration/ReleaseResponse	Response to "Release"
http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerationEnd	Notifies that an enumerator has terminated (not required in WS-Management)
http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe	Models a subscription to an event source
http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscribeResponse	Response to "Subscribe"
http://schemas.xmlsoap.org/ws/2004/08/eventing/Renew	Renews a subscription prior to its expiration
http://schemas.xmlsoap.org/ws/2004/08/eventing/RenewResponse	Response to "Renew"
http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatus	Requests the status of a subscription
http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatusResponse	Response to "GetStatus"
http://schemas.xmlsoap.org/ws/2004/08/eventing/Unsubscribe	Removes an active subscription
http://schemas.xmlsoap.org/ws/2004/08/eventing/UnsubscribeResponse	Response to "Unsubscribe"
http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscriptionEnd	Delivers a message to indicate that a subscription has terminated
http://schemas.dmtf.org/wbem/wsman/1/wsman/Events	Delivers batched events based on a subscription
http://schemas.dmtf.org/wbem/wsman/1/wsman/Heartbeat	A pseudo-event that models a heartbeat of an active subscription; delivered when no real events are available, but used to indicate that the event subscription and delivery mechanism is still active
http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents	A pseudo-event that indicates that the real event was dropped
http://schemas.dmtf.org/wbem/wsman/1/wsman/Ack	Used by event subscribers to acknowledge receipt of events; allows event streams to be strictly sequenced
http://schemas.dmtf.org/wbem/wsman/1/wsman/Event	Used for a singleton event that does not define its own action

1680 **R5.4.6.5-4:** A custom action may be supported if the operation is a custom method whose
1681 semantic meaning is not present in the table.

1682 **R5.4.6.5-5:** All notifications shall contain a unique action URI that identifies the type of the event
1683 delivery. For singleton notifications with only one event per message (the delivery mode
1684 <http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push>), the `wsa:Action` URI
1685 defines the event type. For other delivery modes, the Action varies, as described in clause 10.2.7
1686 of this specification.

1687 **5.4.6.6 wsa:From**

1688 The `wsa:From` header can be used in any messages, responses, or events to indicate the source.
1689 When the same connection is used for both request and reply, this header provides no useful
1690 information, but can be useful in cases where the response arrives on a different connection.

1691 **R5.4.6.6-1:** A conformant service may include a `wsa:From` address in the message. A
1692 conformant service should process any incoming message that has a `wsa:From` element.

1693 **R5.4.6.6-2:** A conformant service should not fault any message with a `wsa:From` element,
1694 regardless of whether the `mustUnderstand` attribute is included.

1695 NOTE: Processing the `wsa:From` header is trivial because it has no effect on the meaning of the
1696 message. The *From* address is primarily for auditing and logging purposes.

1697 **6 WS-Management Control Headers**

1698 WS-Management defines several SOAP headers that can be used with any operation.

1699 **6.1 wsman:OperationTimeout**

1700 Most management operations are time-critical due to quality-of-service constraints and obligations. If
1701 operations cannot be completed in a specified time, the service returns a fault so that a client can
1702 comply with its obligations. The following header value can be supplied with any WS-Management
1703 message to indicate that the client expects a response or a fault within the specified time:

```
1704 (1) <wsman:OperationTimeout> xs:duration </wsman:OperationTimeout>
```

1705 **R6.1-1:** All request messages may contain a `wsman:OperationTimeout` header element that
1706 indicates the maximum amount of time the client is willing to wait for the service to issue a
1707 response. The service should interpret the timeout countdown as beginning from the point the
1708 message is processed until a response is generated.

1709 **R6.1-2:** The service should *immediately* issue a `wsman:TimedOut` fault if the countdown time is
1710 exceeded and the operation is not yet complete. If the `OperationTimeout` value is not valid, a
1711 `wsa:InvalidMessageInformationHeader` fault should be returned.

1712 **R6.1-3:** If the service does not support user-defined timeouts, a `wsman:UnsupportedFeature`
1713 fault should be returned with the following detail code:

```
1714 http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OperationTimeout
```

1715 **R6.1-4:** If the `wsman:OperationTimeout` element is omitted, the service may interpret this
1716 omission as an instruction to block indefinitely until a response is available, or it may impose a
1717 default timeout.

1718 These rules do not preclude services from supporting infinite or very long timeouts. Because network
1719 connections seldom block indefinitely with no traffic occurring, some type of transport timeout is likely.
1720 Also the countdown is initiated from the time the message is received, so network latency is not
1721 included. If a client needs to discover the range of valid timeouts or defaults, metadata can be
1722 retrieved, but the format of such metadata is beyond the scope of this specification.

1723 If the timeout occurs in such a manner that the service has already performed some of the work
1724 associated with the request, the service state reaches an anomalous condition. This specification
1725 does not attempt to address behavior in this situation. Clearly, services can attempt to undo the
1726 effects of any partially complete operations, but this is not always practical. In such cases, the service
1727 can keep a local log of requests and operations, which the client can query later.

1728 For example, if a Delete operation is in progress and a timeout occurs, the service decides whether to
1729 attempt a rollback or roll-forward of the deletion, even though it issues a wsman:TimedOut fault. The
1730 service can elect to include additional information in the fault (see 14.5) regarding its internal policy in
1731 this regard. The service can attempt to return to the state that existed before the operation was
1732 attempted, but this is not always possible.

1733 **R6.1-5:** If the mustUnderstand attribute is applied to the wsman:OperationTimeout element and
1734 the service understands wsman:OperationTimeout, the service shall observe the requested value
1735 or return the fault specified in R6.1-2. The service should attempt to complete the request within
1736 the specified time or issue a fault without any further delay.

1737 Clients can always omit the mustUnderstand header for uniform behavior against all implementations.
1738 It is not an error for a compliant service to ignore the timeout value or treat it as a hint if
1739 mustUnderstand is omitted.

1740 EXAMPLE: The following is an example of a correctly formatted 30-second timeout in the SOAP header:

```
1741 (1) <wsman:OperationTimeout>PT30S</wsman:OperationTimeout>
```

1742 If the transport timeout occurs before the actual wsman:OperationTimeout, the operation can be
1743 treated as specified in 13.3, the same as a failed connection. In practice, the network transport
1744 timeout can be configured to be longer than any expected wsman:OperationTimeout.

1745 6.2 wsman:MaxEnvelopeSize

1746 To prevent a response beyond the capability of the client, the request message can contain a
1747 restriction on the response size.

1748 The following header value may be supplied with any WS-Management message to indicate that the
1749 client expects a response whose total SOAP envelope does not exceed the specified number of
1750 octets:

```
1751 (1) <wsman:MaxEnvelopeSize> xs:positiveInteger </wsman:MaxEnvelopeSize>
```

1752 The limitation is on the entire envelope. Resource-constrained implementations need a reliable figure
1753 for the required amount of memory for all SOAP processing, not just the SOAP Body.

1754 **R6.2-1:** All request messages may contain a wsman:MaxEnvelopeSize header element that
1755 indicates the maximum number of octets (not characters) in the entire SOAP envelope in the
1756 response. If the service cannot compose a reply within the requested size, it should return a
1757 wsman:EncodingLimit fault with the following detail code:

```
1758 http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize
```

1759 **R6.2-2:** If the mustUnderstand attribute is set to “true”, the service shall comply with the
1760 request. If the response would exceed the maximum size, the service should return a
1761 wsman:EncodingLimit fault. Because a service might execute the operation prior to knowing the

1762 response size, the service should undo any effects of the operation before issuing the fault. If the
1763 operation cannot be reversed (such as a destructive Put or Delete, or a Create), the service shall
1764 indicate that the operation succeeded in the wsman:EncodingLimit fault with the following detail
1765 code:

1766 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnreportableSuccess>

1767 **R6.2-3:** If the mustUnderstand attribute is set to “false”, the service may ignore the header.

1768 **R6.2-4:** Services should reject any MaxEnvelopeSize value less than 8192 octets. This number
1769 is the safe minimum in which faults can be reliably encoded for all character sets. If the requested
1770 size is less than this, the service should return a wsman:EncodingLimit fault with the following
1771 detail code:

1772 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MinimumEnvelopeLimit>

1773 A service might have its own encoding limit independent of what the client specifies, and the same
1774 fault applies.

1775 **R6.2-5:** If the service cannot compose a reply within its own internal limits, the service should
1776 return a wsman:EncodingLimit fault with the following detail code:

1777 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ServiceEnvelopeLimit>

1778 The definition of the wsman:MaxEnvelopeSize element in the schema contains a Policy attribute
1779 because this element is used for other purposes. This specification does not define a meaning for the
1780 Policy attribute when the wsman:MaxEnvelopeSize element is used as a SOAP header.

1781 **R6.2-6:** Clients should not add the Policy attribute to the wsman:MaxEnvelopeSize element
1782 when it is used as a SOAP header. Services should ignore the Policy attribute if it appears in the
1783 wsman:MaxEnvelopeSize element when used as a SOAP header.

1784 **6.3 wsman:Locale**

1785 Management operations often span locales, and many items in responses can require translation.
1786 Typically, translation is required for descriptive information, intended for human readers, that is sent
1787 back in the response. If the client requires such output to be translated into a specific language, it can
1788 employ the optional wsman:Locale header, which makes use of the standard XML attribute xml:lang,
1789 as follows:

1790 (1) `<wsman:Locale xml:lang="xs:language" s:mustUnderstand="false"/>`

1791 **R6.3-1:** If the mustUnderstand attribute is omitted or set to “false”, the service should use this
1792 value when composing the response message and adjust any localizable values accordingly.
1793 This use is recommended for most cases. The locale is treated as a hint in this case.

1794 **R6.3-2:** If the mustUnderstand attribute is set to “true”, the service shall ensure that the replies
1795 contain localized information where appropriate, or else the service shall issue a
1796 wsman:UnsupportedFeature fault with the following detail code:

1797 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Locale>

1798 A service may always fault if wsman:Locale contains s:mustUnderstand set to “true”, because it
1799 may not be able to ensure that the reply is localized.

1800 Some implementations delegate the request to another subsystem for processing, so the service
1801 cannot be certain that the localization actually occurred.

1802 **R6.3-3:** The value of the `xml:lang` attribute in the `wsman:Locale` header shall be a valid [RFC](#)
1803 [5646](#) language code.

1804 **R6.3-4:** In any response, event, or singleton message, the service should include the `xml:lang`
1805 attribute in the `s:Envelope` (or other elements) to signal to the receiver that localized content
1806 appears in the body of the message. This attribute may be omitted if no descriptive content
1807 appears in the body. Including the `xml:lang` attribute is not an error, even if no descriptive content
1808 occurs.

1809 EXAMPLE:

```
1810 (1) <s:Envelope  
1811 (2)   xml:lang="en-us"  
1812 (3)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"  
1813 (4)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"  
1814 (5)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">  
1815 (6)   <s:Header> ... </s:Header>  
1816 (7)   <s:Body> ... </s:Body>  
1817 (8) </s:Envelope>
```

1818 The `xml:lang` attribute can appear on any content in the message, although a simpler approach
1819 allows the client always to check for the attribute in one place, the `s:Envelope` wrapper.

1820 **R6.3-5:** For operations that span multiple message sequences, the `wsman:Locale` element is
1821 processed in the initial message only. It should be ignored in subsequent messages because the
1822 first message establishes the required locale. The service may issue a fault if the `wsman:Locale`
1823 is present in subsequent messages and the value is different from that used in the initiating
1824 request.

1825 This rule applies primarily to Enumerate and Pull messages. The locale is clearly established during
1826 the initial Enumerate request, so changing the locale during the enumeration serves no purpose. The
1827 service ignores any `wsman:Locale` elements in subsequent Pull messages, but the client can ensure
1828 that the value does not change between Pull requests. This uniformity enables the client to construct
1829 messages more easily.

1830 It is recommended (as established in R6.3-1) that the `wsman:Locale` element never contain a
1831 `mustUnderstand` attribute. In this way, the client will not receive faults in unexpected places.

1832 6.4 wsman:OptionSet

1833 The `OptionSet` header is used to pass a set of switches to the service to modify or refine the nature of
1834 the request. This facility is intended to help the service observe any context or side effects desired by
1835 the client, but *not* to alter the output schema or modify the meaning of the addressing. Options are
1836 similar to switches used in command-line shells in that they are service-specific, text-based
1837 extensions.

1838 **R6.4-1:** Any request message may contain a `wsman:OptionSet` header, which wraps a set of
1839 optional switches or controls on the message. These switches help the service compose the
1840 desired reply or observe the required side effect.

1841 **R6.4-2:** The service should not send responses, unacknowledged events, or singleton
1842 messages that contain `wsman:OptionSet` headers unless it is acting in the role of a client to
1843 another service. Those headers are intended for request messages to which a subsequent
1844 response is expected, including acknowledged events.

1845 **R6.4-3:** If the `mustUnderstand` attribute is omitted from the `OptionSet` block or if it is present
1846 with a value of "false", the service may ignore the entire `wsman:OptionSet` block. If it is present

- 1847 with a value of "true" and the service does not support wsman:OptionSet, the service shall return
1848 a s:NotUnderstood fault.
- 1849 Services can process an OptionSet block if it is present, but they are not required to understand or
1850 process individual options, as shown in R6.4-6. However, if MustComply is set to "true" on any given
1851 option, then mustUnderstand needs to be set to "true". Doing so avoids the incongruity of allowing the
1852 entire OptionSet block to be ignored while having MustComply on individual options.
- 1853 **R6.4-4:** Each resource class may observe its own set of options, and an individual instance of
1854 that resource class may further observe its own set of options. Consistent option usage is not
1855 required across resource class and instance boundaries. The metadata formats and definitions of
1856 options are beyond the scope of this specification and may be service-specific.
- 1857 **R6.4-5:** Any number of individual option elements may appear under the wsman:OptionSet
1858 wrapper. Option names may be repeated if appropriate. The content shall be a simple string
1859 (xs:string). This specification places no restrictions on whether the names or values are to be
1860 treated in a case-sensitive or case-insensitive manner. However, case usage shall be retained as
1861 the message containing the OptionSet element and its contents are propagated through SOAP
1862 intermediaries.
- 1863 Interpretation of the option with regard to case sensitivity is up to the service and the definition of the
1864 specific option because the value might be passed through to real-world subsystems that
1865 inconsistently expose case usage. Where interoperability is a concern, the client can omit both
1866 mustUnderstand and MustComply attributes.
- 1867 **R6.4-6:** Individual option values may be advisory or may be required by the client. The service
1868 shall observe and execute any option marked with the MustComply attribute set to "true", or
1869 return a wsman:InvalidOptions fault with the following detail code:
1870 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/NotSupported>
- 1871 Any option not marked with this attribute (or if the attribute is set to "false") is advisory to the
1872 service, and the service may ignore it. If any option is marked with MustComply set to "true", then
1873 the mustUnderstand attribute shall be used on the entire wsman:OptionSet block.
- 1874 This capability is required when the service delegates interpretation and execution of the options
1875 to another component. In many cases, the SOAP processor cannot know if the option was
1876 observed and can only pass it along to the next subsystem.
- 1877 **R6.4-7:** Options may optionally contain a Type attribute, which indicates the data type of the
1878 content of the Option element. A service may require that this attribute be present on any given
1879 option and that it be set to the QName of a valid XML schema data type. Only the standard
1880 simple types declared in the <http://www.w3.org/2001/XMLSchema> namespace are supported in
1881 this version of WS-Management.
- 1882 This rule can help some services distinguish numeric or date/time types from other string values.
- 1883 **R6.4-8:** Options should not be used as a replacement for the documented parameterization
1884 technique for the message; they should be used only as a modifier for it.
- 1885 Options are primarily used to establish context or otherwise instruct the service to perform side-band
1886 operations while performing the operation, such as turning on logging or tracing.
- 1887 **R6.4-9:** The following faults should be returned by the service:
- 1888 • when options are not supported, **wsman:InvalidOptions** with the following detail code:
1889 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/NotSupported>

- 1890 • when one or more option names are not valid or supported by the specific
1891 resource, **wsman:InvalidOptions** with the following detail code:
- 1892 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidName>
- 1893 • when the value is not correct for the option name, **wsman:InvalidOptions** with the
1894 following detail code:
- 1895 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValue>

1896 **R6.4-10:** For operations that span multiple message sequences, the wsman:OptionSet element
1897 is processed in the initial message only. It should be ignored in subsequent messages because
1898 the first message establishes the required set of options. The service may issue a fault if the
1899 wsman:OptionSet is present in subsequent messages and the value is different from that used in
1900 the initiating request, or the service may ignore the values of wsman:OptionSet in such
1901 messages.

1902 This rule applies primarily to Enumerate and Pull messages. The set of options is established once
1903 during the initial Enumerate request, so changing the options during the enumeration would constitute
1904 an error.

1905 Options are intended to make operations more efficient or to preprocess output on behalf of the client.
1906 For example, the options could indicate to the service that the returned values are to be recomputed
1907 and that cached values are not to be used, or that any optional values in the reply may be omitted.
1908 Alternately, the options could be used to indicate verbose output within the limits of the XML schema
1909 associated with the reply.

1910 Option values are not intended to contain XML. If XML-based input is required, a custom operation
1911 with its own wsa:Action URI is the correct model for the operation. This ensures that no backdoor
1912 parameters are introduced over well-known message types. For example, when issuing a Subscribe
1913 request, the message already defines a technique for passing an event filter to the service, so the
1914 option is not used to circumvent this and pass a filter using an alternate method.

1915 **EXAMPLE:** The following is an example of wsman:OptionSet:

```

1916 (1) <s:Envelope
1917 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1918 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1919 (4)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
1920 (5)   xmlns:xs="http://www.w3.org/2001/XMLSchema">
1921 (6)   <s:Header>
1922 (7)     ...
1923 (8)     <wsman:OptionSet s:mustUnderstand="true">
1924 (9)       <wsman:Option Name="VerbosityLevel" Type="xs:int">
1925 (10)        3
1926 (11)      </wsman:Option>
1927 (12)      <wsman:Option Name="LogAllRequests" MustComply="true"/>
1928 (13)    </wsman:OptionSet>
1929 (14)    ...
1930 (15)  </s:Header>
1931 (16)  <s:Body> ... </s:Body>
1932 (17) </s:Envelope>

```

1933 The following definitions provide additional, normative constraints on the preceding outline:

1934 wsman:OptionSet

1935 used to wrap individual option blocks

1936 In this example, s:mustUnderstand is set to "true", indicating that the client is requiring the
1937 service to process the option block using the given rules.

- 1938 wsman:OptionSet/wsman:Option/@Name
 1939 identifies the option (an xs:string), which may be a simple name or a URI
 1940 This name is scoped to the resource to which it applies. The name may be repeated in
 1941 subsequent elements. The name cannot be blank and can be a short non-colliding URI that is
 1942 vendor-specific.
- 1943 wsman:OptionSet/wsman:Option/@MustComply
 1944 if set to "true", indicates that the option shall be observed; otherwise, indicates an advisory or a
 1945 hint
- 1946 wsman:OptionSet/wsman:Option/@Type
 1947 (optional) if present, indicates the data type of the element content, which helps the service to
 1948 interpret the content
 1949 A service may require this attribute to be present on any given option element.
- 1950 wsman:OptionSet/wsman:Option
 1951 the content of the option
 1952 The value may be any simple string value. If the option value is empty, the option should be
 1953 interpreted as logically "true", and the option should be "enabled". The following example
 1954 enables the "Verbose" option:
 1955

```
(1) <wsman:Option Name="Verbose"/>
```
- 1956 Options are logically false if they are not present in the message. All other cases require an explicit
 1957 string to indicate the option value. The reasoning for allowing the same option to repeat is to allow
 1958 specification of a list of options of the same name.

1959 6.5 wsman:RequestEPR

1960 Some service operations, including "Put", are able to modify the resource representation in such a
 1961 way that the update results in a logical identity change for the resource, such as the "rename" of a
 1962 document. In many cases, this modification in turn alters the EPR of that resource after the operation
 1963 is completed, as EPRs are often dynamically derived from naming values within the resource
 1964 representation itself. This behavior is common in SOAP implementations that delegate operations to
 1965 underlying systems.

1966 To provide the client a way to determine when such a change has happened, two SOAP headers are
 1967 defined to request and return the EPR of a resource instance.

1968 In any WS-Management request message, the following header may appear:

1969

```
(1) <wsman:RequestEPR .../>
```

1970 **R6.5-1:** A service receiving a message that contains the wsman:RequestEPR header block
 1971 should return a response that contains a wsman:RequestedEPR header block. This block
 1972 contains the most recent EPR of the resource being accessed or a status code if the service
 1973 cannot determine or return the EPR. This EPR reflects any identity changes that may have
 1974 occurred as a result of the current operation, as set forth in the following behavior. The header
 1975 block in the corresponding response message has the following format:

1976

```
(1) <wsman:RequestedEPR ...>  

  1977 (2) [ <wsa:EndpointReference>  

  1978 (3) wsa:EndpointReferenceType  

  1979 (4) </wsa:EndpointReference> |  

  1980 (5) <wsman:EPRInvalid/> |  

  1981 (6) <wsman:EPRUnknown/> ]  

  1982 (7) </wsman:RequestedEPR>
```

1983 The following definitions describe additional, normative constraints on the preceding format:

1984 wsman:RequestedEPR/wsa:EndpointReference

1985 one of three elements that can be returned as a child element of the wsman:RequestedEPR
1986 element

1987 The use of this element indicates that the service understood the request to return the EPR of
1988 the resource and is including the EPR of the resource. The returned EPR is calculated after all
1989 intentional effects or side effects of the associated request message have occurred. The EPR
1990 may not have changed as a result of the operation, but the service is still obligated to return it.

1991 wsman:RequestedEPR/wsman:EPRInvalid

1992 one of three elements that can be returned as a child element of the wsman:RequestedEPR
1993 element

1994 The use of this element (no value is required) indicates that the service understands the request
1995 to return the EPR of the resource but is unable to calculate a full EPR. However, the service is
1996 able to determine that this message exchange has modified the resource representation in such
1997 a way that any previous references to the resource are no longer valid. When EPRInvalid is
1998 returned, the client shall not use the old wsa:EndpointReference in subsequent operations.

1999 wsman:RequestedEPR/wsman:EPRUnknown

2000 one of three elements that can be returned as a child element of the wsman:RequestedEPR
2001 element

2002 The use of this element (no value is required) indicates that the service understands the request
2003 to return the EPR of the resource but is unable to determine whether existing references to the
2004 resource are still valid. When EPRUnknown is returned, the client may attempt to use the old
2005 wsa:EndpointReference in subsequent operations. The result of using an old
2006 wsa:EndpointReference, however, is unpredictable; a result may be a fault or a successful
2007 response.

2008 **7 Resource Access**

2009 **7.1 General**

2010 Resource access applies to all synchronous operations regarding getting, setting, and enumerating
2011 values. The subclauses in clause 7 define a mechanism for acquiring management-specific XML-
2012 based representations of entities using the Web service infrastructure, such as managed resources.

2013 Specifically, two operations are defined for sending and receiving the management representation of
2014 a given resource and two operations are defined for creating and deleting a management resource
2015 and its corresponding representation. Multi-instance retrieval is achieved using the enumeration
2016 messages. This specification does not define any messages or techniques for batched operations,
2017 such as batched Get or Delete. All such operations can be sent as a series of single messages.

2018 It should be noted that the state maintenance of a resource is at most subject to the "best efforts" of
2019 the hosting server. When a client receives the server's acceptance of a request to create or update a
2020 resource, it can reasonably expect that the resource now exists at the confirmed location and with the
2021 confirmed representation, but this is not a guarantee, even in the absence of any third parties. The
2022 server may change the representation of a resource, may remove a resource entirely, or may bring
2023 back a resource that was deleted.

2024 For instance, the server may store resource state information on a disk drive. If that drive crashes and
2025 the server recovers state information from a backup tape, changes that occurred after the backup
2026 was made would be lost.

2027 A server may have other operational processes that change resource state information. A server may
 2028 run a background process that examines resources for objectionable content and deletes any such
 2029 resources it finds. A server may purge resources that have not been accessed for some period of
 2030 time. A server may apply storage quotas that cause it to occasionally purge resources.

2031 In essence, the confirmation by a service of having processed a request to create, modify, or delete a
 2032 resource implies a commitment only at the instant that the confirmation was generated. While the
 2033 usual case should be that resources are long-lived and stable, there are no guarantees, and clients
 2034 should code defensively.

2035 There is no requirement for uniformity in resource representations between the messages defined in
 2036 this specification. For example, the representations required by Create or Put may differ from the
 2037 representation returned by Get, depending on the semantic requirements of the service. Additionally,
 2038 there is no requirement that the resource content is fixed for any given endpoint reference. The
 2039 resource content may vary based on environmental factors, such as the security context, time of day,
 2040 configuration, or the dynamic state of the service.

2041 As per the SOAP processing model, other specifications may define SOAP headers that may be
 2042 optionally added to request messages to require the transfer of subsets or the application of
 2043 transformations of the resource associated with the endpoint reference. When the Action URIs
 2044 defined by this specification are used, such extension specifications must also allow the basic
 2045 processing models defined herein.

2046 NOTE: The WSDL for the resource access operations (see ANNEX G), as well as the pseudo schema and
 2047 example message fragments throughout clause 7, is not usable as represented without first replacing the
 2048 "*resource-specific-GED*" text with the application-defined GED.

2049 EXAMPLE 1: Following is a full example of a hypothetical Get request:

```

2050 (1) <s:Envelope
2051 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2052 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2053 (4)   xmlns:wsmn="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
2054 (5)   <s:Header>
2055 (6)     <wsa:To>http://1.2.3.4/wsman/</wsa:To>
2056 (7)     <wsman:ResourceURI>http://example.org/2005/02/physicalDisk
2057 (8)       </wsman:ResourceURI>
2058 (9)     <wsa:ReplyTo>
2059 (10)      <wsa:Address>
2060 (11)        http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
2061 (12)      </wsa:Address>
2062 (13)    </wsa:ReplyTo>
2063 (14)    <wsa:Action>
2064 (15)      http://schemas.xmlsoap.org/ws/2004/09/transfer/Get
2065 (16)    </wsa:Action>
2066 (17)    <wsa:MessageID>
2067 (18)      urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a87
2068 (19)    </wsa:MessageID>
2069 (20)    <wsman:SelectorSet>
2070 (21)      <wsman:Selector Name="LUN"> 2 </wsman:Selector>
2071 (22)    </wsman:SelectorSet>
2072 (23)    <wsman:OperationTimeout> PT30S </wsman:OperationTimeout>
2073 (24)  </s:Header>
2074 (25)  <s:Body/>
2075 (26) </s:Envelope>
  
```

2076 Notice that the `wsa:ReplyTo` indicates the response is to be sent on the same connection as the
 2077 request (line 10), the action is a Get (line 14), and the ResourceURI (line 7) and `wsman:SelectorSet`
 2078 (line 20) are used to address the requested management information. This example assumes that the

2079 WS-Management default addressing model is in use. The service is expected to complete the
2080 operation in 30 seconds or return a fault to the client (line 22).

2081 Also, the s:Body in a Get request has no content.

2082 EXAMPLE 1 (continued):The following shows a hypothetical response to the preceding hypothetical Get request:

```

2083 (26) <s:Envelope
2084 (27)     xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2085 (28)     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2086 (29)     xmlns:wsmn="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
2087 (30)   <s:Header>
2088 (31)     <wsa:To>
2089 (32)       http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
2090 (33)     </wsa:To>
2091 (34)     <wsa:Action s:mustUnderstand="true">
2092 (35)       http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse
2093 (36)     </wsa:Action>
2094 (37)     <wsa:MessageID s:mustUnderstand="true">
2095 (38)       urn:uuid:217a431c-b071-3301-9bb8-5f538bec89b8
2096 (39)     </wsa:MessageID>
2097 (40)     <wsa:RelatesTo>
2098 (41)       urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a87
2099 (42)     </wsa:RelatesTo>
2100 (43)   </s:Header>
2101 (44)   <s:Body>
2102 (45)     <PhysicalDisk
2103 (46)       xmlns="http://schemas.example.org/2005/02/samples/physDisk">
2104 (47)       <Manufacturer> Acme, Inc. </Manufacturer>
2105 (48)       <Model> 123-SCSI 42 GB Drive </Model>
2106 (49)       <LUN> 2 </LUN>
2107 (50)       <Cylinders> 16384 </Cylinders>
2108 (51)       <Heads> 80 </Heads>
2109 (52)       <Sectors> 63 </Sectors>
2110 (53)       <OctetsPerSector> 512 </OctetsPerSector>
2111 (54)       <BootPartition> 0 </BootPartition>
2112 (55)     </PhysicalDisk>
2113 (56)   </s:Body>
2114 (57) </s:Envelope>

```

2115 Notice that the response uses the wsa:To address (line 32) that the original request had specified in
2116 wsa:ReplyTo. Also, the wsa:MessageID for this response is unique (line 38). The wsa:RelatesTo
2117 (line 41) contains the UUID of the wsa:MessageID of the original request to allow the client to
2118 correlate the response.

2119 The s:Body (lines 44-55) contains the requested resource representation.

2120 The same general approach exists for Delete, except that no content exists in the response s:Body.
2121 The Create and Put operations are similar, except that they contain content in the request s:Body to
2122 specify the values being created or updated.

2123 7.2 Addressing Uniformity

2124 Where practical, the EPR of the resource can be the same whether a Get, Delete, or Put operation is
2125 being used. This is not a strict requirement, but it reduces the education and training required to
2126 construct and use WS-Management-aware tools.

2127 Create is a special case, in that the EPR of the newly created resource is often not known until the
 2128 resource is actually created. For example, although it might be possible to return running process
 2129 information using a hypothetical *ProcessID* in an addressing header, it is typically not possible to
 2130 assert the *ProcessID* during the creation phase because the underlying system does not support the
 2131 concept. Thus, the Create operation would not have the same addressing headers as the
 2132 corresponding Get or Delete operations.

2133 If the WS-Management default addressing model is in use, it would be typical to use the
 2134 ResourceURI as a "type" and selector values for "instance" identification. Thus, the same address
 2135 would be used for Get, Put, and Delete when working with the same instance. When enumerating all
 2136 instances, the selectors would be omitted and the ResourceURI would be used alone to indicate the
 2137 "type" of the object being enumerated. The Create operation might also share this usage, or have its
 2138 own ResourceURI and selector usage (or not even use selectors). This pattern is not a requirement.

2139 Throughout, it is expected that the s:Body of the messages contains XML with correct and valid XML
 2140 namespaces referring to XML Schemas that can validate the message. Most services and clients do
 2141 not perform real-time validation of messages in production environments because of performance
 2142 constraints; however, during debugging or other systems verification, validation might be enabled,
 2143 and messages without the appropriate XML namespace declarations would be considered invalid.

2144 When performing resource access operations, side effects might occur. For example, deletion of a
 2145 particular resource by using Delete can result in several other dependent instances disappearing, and
 2146 a Create operation can result in the logical creation of more than one resource that can be
 2147 subsequently returned through a Get operation. Similarly, a Put operation can result in a rename of
 2148 the target instance, a rename of some unrelated instance, or the deletion of some unrelated instance.
 2149 These side effects are service specific, and this specification makes no statements about the
 2150 taxonomy and semantics of objects over which these operations apply.

2151 7.3 Get

2152 A Web service operation (Get) is defined for fetching a one-time snapshot of the representation of a
 2153 resource. A snapshot is a complete XML representation of a resource at the time the service
 2154 processes the request.

2155 The Get request message shall be of the following form:

```

2156 (1) <s:Envelope ...>
2157 (2)   <s:Header ...>
2158 (3)     <wsa:Action>
2159 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/Get
2160 (5)     </wsa:Action>
2161 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
2162 (7)     <wsa:To>xs:anyURI</wsa:To>
2163 (8)     ...
2164 (9)   </s:Header>
2165 (10)  <s:Body .../>
2166 (11) </s:Envelope>
  
```

2167 The following describes additional, normative constraints on the preceding outline:

2168 /s:Envelope/s:Header/wsa:Action

2169 This required element shall contain the value
 2170 http://schemas.xmlsoap.org/ws/2004/09/transfer/Get. If a SOAP Action URI is also present in the
 2171 underlying transport, its value shall convey the same value.

2172 A Get request shall be targeted at the resource whose representation is desired.

2173 There are no body blocks defined by default for a Get Request. As per the SOAP processing model,
 2174 other specifications may introduce various types of extensions to the semantics of this message that

2175 are enabled through headers tagged with `s:mustUnderstand="true"`. Such extensions may define how
 2176 resource or subsets of it are to be retrieved or transformed prior to retrieval. Specifications that define
 2177 such extensions shall allow processing the basic Get request message without those extensions.
 2178 Because the response may not be sent to the original sender, extension specifications should
 2179 consider adding a corresponding SOAP header value in the response to signal to the receiver that the
 2180 extension is being used.

2181 Implementations may respond with a fault message using the standard fault codes defined in
 2182 Addressing (for example, `wsa:ActionNotSupported`). Other components of the preceding outline are
 2183 not further constrained by this specification.

2184 If the resource accepts a Get request, it shall reply with a response of the following form:

```

2185 (1) <s:Envelope ...>
2186 (2)   <s:Header ...>
2187 (3)     <wsa:Action>
2188 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse
2189 (5)     </wsa:Action>
2190 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
2191 (7)     <wsa:To>xs:anyURI</wsa:To>
2192 (8)     ...
2193 (9)   </s:Header>
2194 (10)  <s:Body ...>
2195 (11)   resource-specific-element
2196 (12)  </s:Body>
2197 (13) </s:Envelope>
  
```

2198 The following describes additional, normative constraints on the preceding outline:

2199 `/s:Envelope/s:Header/wsa:Action`

2200 This required element shall contain the value
 2201 `http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse`. If a SOAP Action URI is also
 2202 present in the underlying transport, its value shall convey the same value.

2203 `/s:Envelope/s:Body/child`

2204 The representation itself shall be the child element of the `SOAP:Body` element of the response
 2205 message.

2206 Other components of the preceding outline are not further constrained by this specification.

2207 The Get operation retrieves resource representations. The message can be targeted to return a
 2208 complex XML document or to return a single, simple value. The nature and complexity of the
 2209 representation is not constrained by this specification.

2210 **R7.3-1:** A conformant service should support Get operations to service metadata requests
 2211 about the service itself or to verify the result of a previous action or operation.

2212 This statement does not constrain implementations from supplying additional similar methods for
 2213 resource and metadata retrieval.

2214 **R7.3-2:** Execution of Get should not in itself have side effects on the value of the resource.

2215 **R7.3-3:** If an object cannot be retrieved due to locking conditions, simultaneous access, or
 2216 similar conflicts, a `wsman:Concurrency` fault should be returned.

2217 In practice, Get is designed to return XML that corresponds to real-world objects. To retrieve
 2218 individual property values, either the client can postprocess the XML content for the desired value, or
 2219 the service can support fragment-level access (7.7).

2220 Fault usage is generally as described in clause 14. An inability to locate or access the resource is
 2221 equivalent to problems with the SOAP message when the EPR is defective. There are no "Get-
 2222 specific" faults.

2223 7.4 Put

2224 A Web service operation (Put) is defined for updating a resource by providing a replacement
 2225 representation. A resource may accept updates that provide different XML representations than that
 2226 returned by the resource; in such a case, the semantics of the update operation is defined by the
 2227 resource.

2228 The Put request message shall be of the following form:

```

2229 (1) <s:Envelope ...>
2230 (2)   <s:Header ...>
2231 (3)     <wsa:Action>
2232 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/Put
2233 (5)     </wsa:Action>
2234 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
2235 (7)     <wsa:To>xs:anyURI</wsa:To>
2236 (8)     ...
2237 (9)   </s:Header>
2238 (10)  <s:Body...>
2239 (11)   resource-specific-element
2240 (12)  </s:Body>
2241 (13) </s:Envelope>
  
```

2242 The following describes additional, normative constraints on the preceding outline:

2243 /s:Envelope/s:Header/wsa:Action

2244 This required element shall contain the value
 2245 http://schemas.xmlsoap.org/ws/2004/09/transfer/Put. If a SOAP Action URI is also present in the
 2246 underlying transport, its value shall convey the same value.

2247 /s:Envelope/s:Body/child

2248 The representation to be used for the update shall be the child element of the s:Body element of
 2249 the request message.

2250 A Put request shall be targeted at the resource whose representation is desired to be replaced. As
 2251 per the SOAP processing model, other specifications may introduce various types of extensions to
 2252 this message, which are enabled through headers tagged with s:mustUnderstand="true". Such
 2253 extensions may require that a full or partial update should be accomplished using symbolic,
 2254 instruction-based, or other methodologies.

2255 Extension specifications may also define extensions to the original Put request, enabled by optional
 2256 SOAP headers, which control the nature of the response (see the information about PutResponse
 2257 later in this clause).

2258 Specifications that define any of these extensions shall allow processing of the Put message without
 2259 such extensions.

2260 In addition to the standard fault codes defined in Addressing, implementations may use the fault code
 2261 wsmt:InvalidRepresentation if the presented representation is invalid for the target resource. Other
 2262 components of the preceding outline are not further constrained by this specification.

2263 A successful Put operation updates the current representation associated with the targeted resource.

2264 If the resource accepts a Put request and performs the requested update, it shall reply with a
 2265 response of the following form:

```

2266 (1) <s:Envelope ...>
2267 (2)   <s:Header ...>
2268 (3)     <wsa:Action>
2269 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/PutResponse
2270 (5)     </wsa:Action>
2271 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
2272 (7)     <wsa:To>xs:anyURI</wsa:To>
2273 (8)     ...
2274 (9)   </s:Header>
2275 (10)  <s:Body ...>
2276 (11)   resource-specific-element ?
2277 (12)  </s:Body>
2278 (13) </s:Envelope>

```

2279 /s:Envelope/s:Header/wsa:Action

2280 This required element shall contain the value
 2281 http://schemas.xmlsoap.org/ws/2004/09/transfer/PutResponse. If a SOAP Action URI is also
 2282 present in the underlying transport, its value shall convey the same value.

2283 /s:Envelope/s:Body/child

2284 An implementation of a service shall choose, in advance, whether to return an empty Body or the
 2285 resulting representation of the resource. This choice shall be explicitly stated in the WSDL, if
 2286 WSDL is provided.

2287 By default, a service shall return the current representation of the resource as the child of the
 2288 s:Body element if the updated representation differs from the representation sent in the Put
 2289 request message.

2290 As an optimization and as a service to the requester, the s:Body element of the response
 2291 message should be empty if the updated representation does not differ from the representation
 2292 sent in the Put request message; that is, if the service accepted the new representation
 2293 verbatim.

2294 Such a response (an empty s:Body) implies that the update request was successful in its entirety
 2295 (assuming no intervening mutating operations are performed). A service may return the current
 2296 representation of the resource as the initial child of the s:Body element even in this case,
 2297 however.

2298 Extension specifications may define extensions to the original Put request, enabled by optional
 2299 header values, in order to optimize the response. In the absence of such headers, the behavior shall
 2300 be as previously described. Specifications that define any of these extensions shall allow processing
 2301 the Put message without such extensions. Because the response may not be sent to the original
 2302 sender, extension specifications should consider adding a corresponding SOAP header value in the
 2303 response to signal to the receiver that the extension is being used.

2304 Other components of the preceding outline are not further constrained by this specification.

2305 If a resource can be updated in its entirety within the constraints of the corresponding XML schema
 2306 for the resource, the service can support the Put operation.

2307 **R7.4-1:** A conformant service may support Put.

2308 **R7.4-2:** If a single resource instance can be updated (within the constraints of its schema) by
 2309 using a SOAP message, and that resource subsequently can be retrieved using Get, a service
 2310 should support updating the resource by using Put. The service may additionally export a custom
 2311 method for updates.

2312 **R7.4-3:** If a single resource instance contains a mix of modifiable and non-modifiable
 2313 properties, the Put message may contain values for both the modifiable and non-modifiable
 2314 properties if the XML content is legal with regard to its XML schema namespace. If the Put

2315 message contains values for modifiable properties, the service shall set these properties to these
 2316 values during the Put operation. If the Put message contains values for non-modifiable properties,
 2317 the service should ignore those values during the Put operation. If none of the properties are
 2318 modifiable, the service should return a wsa:ActionNotSupported fault.

2319 This situation typically happens if a Get operation is performed, a value is altered, and the entire
 2320 updated representation is sent using Put. In this case, any read-only values would still be present.

2321 A complication arises because Put contains the complete new representation for the instance. If the
 2322 resource schema requires the presence of any given value (minOccurs is not zero), it will be supplied
 2323 as part of the Put message, even if it is not being altered from its original value.

2324 **R7.4-4:** If a Put operation specifies a modifiable value as NULL using the xsi:nil attribute, then
 2325 the service shall set the value to NULL.

2326 If the schema definition includes elements that are optional (minOccurs=0), the Put message can omit
 2327 these values. Existing implementations provide two different responses when these elements are
 2328 modifiable (writable). They either set the omitted element's value to NULL or leave the value
 2329 unchanged. Given this reality, the following rules apply:

2330 **R7.4-5:** Any modifiable properties that are optional in the XML schema (that is, minOccurs="0")
 2331 and that are omitted from the Put message shall either be set to a resource-specific default
 2332 value or be left unchanged. Setting to a resource specific default value is recommended.

2333 NOTE 1: Elements not set may have their value changed as a result of other constraints.

2334 NOTE 2: The resource-specific default value is outside the scope of this specification.

2335 To update isolated values without having to supply all values, use the fragment-level resource access
 2336 mechanism described in 7.7.

2337 In short, the s:Body of the Put message complies with the constraints of the associated XML schema.

2338 EXAMPLE 1: For example, assume that Get returns the following information:

```
2339 (1) <s:Body>
2340 (2)   <MyObject xmlns="examples.org/2005/02/MySchema">
2341 (3)     <A> 100 </A>
2342 (4)     <B> 200 </B>
2343 (5)     <C> 100 </C>
2344 (6)   </MyObject>
2345 (7) </s:Body>
```

2346 EXAMPLE 2: The corresponding XML schema has defined A, B, and C as minOccurs=1:

```
2347 (8) <xs:element name="MyObjecct">
2348 (9)   <xs:complexType>
2349 (10)    <xs:sequence>
2350 (11)      <xs:element name="A" type="xs:int" minOccurs="1" maxOccurs="1"/>
2351 (12)      <xs:element name="B" type="xs:int" minOccurs="1" maxOccurs="1"/>
2352 (13)      <xs:element name="C" type="xs:int" minOccurs="1" maxOccurs="1"/>
2353 (14)      ...
2354 (15)    </xs:sequence>
2355 (16)  </xs:complexType>
2356 (17) </xs:element>
```

2357 In this case, the corresponding Put needs to contain all three elements because the schema mandates that all
 2358 three be present. Even if the only value being updated is , the client has to supply all three values. This
 2359 usually means that the client first has to issue a Get to preserve the current values of <A> and <C>, change
 2360 to the desired value, and then write the object using Put. As noted in R7.4-3, the service can ignore attempts to
 2361 update values that are read-only with regard to the underlying real-world object.

2362 **R7.4-6:** A conformant service should support Put using the same EPR as a corresponding Get
2363 or other messages, unless the Put mechanism for a resource is semantically distinct.

2364 **R7.4-7:** If the supplied Body does not have the correct content to update the resource, the
2365 service should return a wsmt:InvalidRepresentation fault and detail codes as follows:

2366 • if any values in the s:Body are not correct:

2367 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValues>

2368 • if any values in the s:Body are missing:

2369 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MissingValues>

2370 • if the wrong XML schema namespace is used and is not recognized by the service:

2371 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace>

2372 **R7.4-8:** If an object cannot be updated because of locking conditions, simultaneous access, or
2373 similar conflicts, the service should return a wsman:Concurrency fault.

2374 **R7.4-9:** A Put operation may result in a change to the EPR for the resource because the values
2375 being updated may in turn cause an identity change.

2376 Because WS-Management services typically delegate the Put to underlying subsystems, the service
2377 might not always be aware of an identity change. Clients can make use of the mechanism in 6.5 to be
2378 informed of EPR changes that may have occurred as a side effect of executing a Put operation.

2379 **R7.4-10:** It is recommended that the service return the new representation in the Put response in
2380 all cases. Knowing whether the actual resulting representation is different from the requested
2381 update is often difficult because resource-constrained implementations may have insufficient
2382 resources to determine the equivalence of the requested update with the actual resulting
2383 representation.

2384 The implication of this rule is that if the new representation is not returned, it precisely matches what
2385 was submitted in the Put message. Because implementations can rarely assure this, they can always
2386 return the new representation.

2387 **R7.4-11:** If the success of an operation cannot be reported as described in this clause because
2388 of encoding limits or other reasons, and it cannot be reversed, the service should return a
2389 wsman:EncodingLimit fault with the following detail code:

2390 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnreportableSuccess>

2391 **R7.4-12:** The Put operation may contain updates of multiple values. The service shall
2392 successfully carry out an update of all the specified values or return the fault that was the cause
2393 of the error. If any fault is returned, the implication is that 0... n -1 values were updated out of n
2394 possible update values.

2395 7.5 Delete

2396 This specification defines one Web service operation (Delete) for deleting a resource in its entirety.

2397 Extension specifications may define extensions to the Delete request, enabled by optional header
2398 values, which specifically control preconditions for the Delete to succeed and which may control the
2399 nature or format of the response. Because the response may not be sent to the original sender,
2400 extension specifications should consider adding a corresponding SOAP header value in the response
2401 to signal to the receiver that the extension is being used.

2402 The Delete request message shall be of the following form:

```

2403 (1) <s:Envelope ...>
2404 (2)   <s:Header ...>
2405 (3)     <wsa:Action>
2406 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/Delete
2407 (5)     </wsa:Action>
2408 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
2409 (7)     <wsa:To>xs:anyURI</wsa:To>
2410 (8)     ...
2411 (9)   </s:Header>
2412 (10)  <s:Body ... />
2413 (11) </s:Envelope>

```

2414 The following describes additional, normative constraints on the preceding outline:

2415 /s:Envelope/s:Header/wsa:Action

2416 This required element shall contain the value
 2417 http://schemas.xmlsoap.org/ws/2004/09/transfer/Delete. If a SOAP Action URI is also present in
 2418 the underlying transport, its value shall convey the same value.

2419 A Delete request shall be targeted at the resource to be deleted.

2420 There are no body blocks defined for a Delete Request.

2421 Implementations may respond with a fault message using the standard fault codes defined in
 2422 Addressing (for example, wsa:ActionNotSupported). Other components of the preceding outline are
 2423 not further constrained by this specification.

2424 A successful Delete operation invalidates the current representation associated with the targeted
 2425 resource.

2426 If the resource accepts a Delete request, it shall reply with a response of the following form:

```

2427 (1) <s:Envelope ...>
2428 (2)   <s:Header ...>
2429 (3)     <wsa:Action>
2430 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/DeleteResponse
2431 (5)     </wsa:Action>
2432 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
2433 (7)     <wsa:To>xs:anyURI</wsa:To>
2434 (8)     ...
2435 (9)   </s:Header>
2436 (10)  <s:Body ... />
2437 (11) </s:Envelope>

```

2438 /s:Envelope/s:Header/wsa:Action

2439 This required element shall contain the value
 2440 http://schemas.xmlsoap.org/ws/2004/09/transfer/DeleteResponse. If a SOAP Action URI is also
 2441 present in the underlying transport, its value shall convey the same value.

2442 By default, there are no s:Body blocks defined for a Delete response. Specifications that define
 2443 extensions for use in the original Delete request that control the format of the response shall allow
 2444 processing the Delete message without such extensions.

2445 Other components of the preceding outline are not further constrained by this specification.

2446 In general, the addressing can be the same as for a corresponding Get operation for uniformity, but
 2447 this is not absolutely required.

2448 **R7.5-1:** A conformant service may support Delete.

2449 **R7.5-2:** A conformant service should support Delete using the same EPR as a corresponding
2450 Get or other messages, unless the deletion mechanism for a resource is semantically distinct.

2451 **R7.5-3:** If deletion is supported and the corresponding resource can be retrieved using Get, a
2452 conformant service should support deletion using Delete. The service may additionally export a
2453 custom action for deletion.

2454 **R7.5-4:** If an object cannot be deleted due to locking conditions, simultaneous access, or
2455 similar conflicts, a wsman:Concurrency fault should be returned.

2456 In practice, Delete removes the resource instance from the visibility of the client and is a *logical*
2457 deletion.

2458 The operation might result in an actual deletion, such as removal of a row from a database table, or it
2459 might simulate deletion by unbinding the representation from the real-world object. Deletion of a
2460 "printer," for example, does not result in literal annihilation of the printer, but simply removes it from
2461 the access scope of the service, or "unbinds" it from naming tables. WS-Management makes no
2462 distinction between literal deletions and logical deletions.

2463 To delete individual property values within an object that, itself, is not to be deleted, either the client
2464 can perform a Put, according to section 7.4 or the service can support fragment-level delete (7.7).

2465 Fault usage is generally as described in clause 14. Inability to locate or access the resource is
2466 equivalent to problems with the SOAP message when the EPR is defective. There are no "Delete-
2467 specific" faults.

2468 7.6 Create

2469 A Web service operation (Create) is defined for creating a resource and providing its initial
2470 representation. In some cases, the initial representation may constitute the representation of a logical
2471 constructor for the resource and may thus differ structurally from the representation returned by Get
2472 or the one required by Put. This difference is because the parameterization requirement for creating a
2473 resource is often distinct from the steady-state representation of the resource. Implementations
2474 should provide metadata that describes the use of the representation and how it relates to the
2475 resource which is created, but such mechanisms are beyond the scope of this specification. The
2476 resource factory that receives a Create request allocates a new resource that is initialized from the
2477 presented representation. The new resource is assigned a service-determined endpoint reference
2478 that is returned in the response message.

2479 The Create request message shall be of the following form:

```
2480 (1) <s:Envelope ...>
2481 (2)   <s:Header ...>
2482 (3)     <wsa:Action>
2483 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/Create
2484 (5)     </wsa:Action>
2485 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
2486 (7)     <wsa:To>xs:anyURI</wsa:To>
2487 (8)     ...
2488 (9)   </s:Header>
2489 (10)  <s:Body ...>
2490 (11)   resource-specific-element
2491 (12)  </s:Body>
2492 (13) </s:Envelope>
```

2493 The following describes additional, normative constraints on the preceding outline:

2494 /s:Envelope/s:Header/wsa:Action

2495 This required element shall contain the value
2496 <http://schemas.xmlsoap.org/ws/2004/09/transfer/Create>. If a SOAP Action URI is also present in
2497 the underlying transport, its value shall convey the same value.

2498 /s:Envelope/s:Body/child

2499 The child element of the s:Body element shall not be omitted. The contents of this element are
2500 service-specific, and may contain the literal initial resource representation, a representation of
2501 the constructor for the resource, or other instructions for creating the resource.

2502 Extension specifications may also define extensions to the original Create request, enabled by
2503 optional SOAP headers, which constrain the nature of the response (see information about the
2504 CreateResponse later in this clause). Similarly, they may require headers that control the
2505 interpretation of the s:Body as part of the resource creation process.

2506 Such specifications shall also allow processing the Create message without such extensions.

2507 A Create request shall be targeted at a resource factory capable of creating the desired new
2508 resource. This factory is distinct from the resource being created (which by definition does not exist
2509 prior to the successful processing of the Create request message).

2510 In addition to the standard fault codes defined in Addressing, implementations may use the fault code
2511 `wsmt:InvalidRepresentation` if the presented representation is invalid for the target resource.

2512 Other components of the preceding outline are not further constrained by this specification.

2513 If the resource factory accepts a Create request, it shall reply with a response of the following form:

```
2514 (1) <s:Envelope ...>
2515 (2)   <s:Header ...>
2516 (3)     <wsa:Action>
2517 (4)       http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse
2518 (5)     </wsa:Action>
2519 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
2520 (7)     <wsa:To>xs:anyURI</wsa:To>
2521 (8)     ...
2522 (9)   </s:Header>
2523 (10)  <s:Body ...>
2524 (11)   <wsmt:ResourceCreated>endpoint-reference</wsmt:ResourceCreated>
2525 (12)  </s:Body>
2526 (13) </s:Envelope>
```

2527 /s:Envelope/s:Header/wsa:Action

2528 This required element shall contain the value
2529 <http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse>. If a SOAP Action URI is also
2530 present in the underlying transport, its value shall convey the same value.

2531 /s:Envelope/s:Body/wsmt:ResourceCreated

2532 This required element shall contain a resource reference for the newly created resource. This
2533 resource reference, represented as an endpoint reference as defined in Addressing, shall
2534 identify the resource for future Get, Put, and Delete operations.

2535 Extension specifications may define extensions to the original Create request, enabled by optional
2536 header values. These headers may override the default behavior if they are marked with
2537 `s:mustUnderstand="true"`. In the absence of such optional headers, the behavior shall be as
2538 described in the previous paragraphs. Because the response may not be sent to the original sender,
2539 extension specifications should consider adding a corresponding SOAP header value in the response
2540 to signal to the receiver that the extension is being used.

2541 Other components of the preceding outline are not further constrained by this specification.

2542 In general, the addressing is not the same as that used for Get or Delete in that the EPR assigned to
2543 a newly created instance for subsequent access is not necessarily part of the XML content used for
2544 creating the resource. Because the EPR is usually assigned by the service or one of its underlying
2545 systems, the CreateResponse contains the applicable EPR of the newly created instance.

2546 **R7.6-1:** A conformant service may support Create.

2547 **R7.6-2:** If a single resource can be created using a SOAP message and that resource can be
2548 subsequently retrieved using Get, then a service should support creation of the resource using
2549 Create. The service may additionally export a custom method for instance creation.

2550 **R7.6-3:** If the supplied SOAP Body does not have the correct content for the resource to be
2551 created, the service should return a wsmt:InvalidRepresentation fault and detail codes as follows:

- 2552 • if one or more values in the <s:Body> were not correct:
2553 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValues`

- 2554 • if one or more values in the <s:Body> were missing:
2555 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MissingValues`

- 2556 • if the wrong XML schema namespace was used and is not recognized by the service:
2557 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace`

2558 **R7.6-4:** A service shall not use Create to modify the value of an existing representation (except
2559 as specified in 7.11). If the targeted object already exists, the service should return a
2560 `wsman:AlreadyExists` fault.

2561 The message body for Create is not required to use the same schema as that returned with a Get
2562 operation for the resource. Often, the values required to create a resource are different from those
2563 retrieved using a Get operation or those used for updates with a Put operation.

2564 If a service needs to support creation of individual values within a representation (fragment-level
2565 creation, array insertion, and so on), it can support fragment-level access (7.7).

2566 **R7.6-5:** The response to a Create message shall contain the new EPR of the created resource
2567 in the ResourceCreated element.

2568 **R7.6-6:** This rule intentionally left blank.

2569 **EXAMPLE:** The following is a hypothetical example of a response for a newly created virtual drive:

```
2570 (1) <s:Envelope
2571 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2572 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2573 (4)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
2574 (5)   xmlns:wsmt="http://schemas.xmlsoap.org/ws/2004/09/transfer">
2575 (6)   <s:Header>
2576 (7)     ...
2577 (8)     <wsa:Action>
2578 (9)       http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse
2579 (10)    </wsa:Action>
2580 (11)    ...
2581 (12)   </s:Header>
2582 (13)   <s:Body>
2583 (14)     <wsmt:ResourceCreated>
2584 (15)     <wsa:Address>
```

```

2585 (16) http://1.2.3.4/wsman/
2586 (17) </wsa:Address>
2587 (18) <wsa:ReferenceParameters>
2588 (19) <wsman:ResourceURI>
2589 (20) http://example.org/2005/02/virtualDrive
2590 (21) </wsman:ResourceURI>
2591 (22) <wsman:SelectorSet>
2592 (23) <wsman:Selector Name="ID"> F: </wsman:Selector>
2593 (24) </wsman:SelectorSet>
2594 (25) </wsa:ReferenceParameters>
2595 (26) </wsmt:ResourceCreated>
2596 (27) </s:Body>
2597 (28) </s:Envelope>

```

2598 This example assumes that the default addressing model is in use. The response contains a ResourceCreated
 2599 block (lines 14-26), which contains the new endpoint reference of the created resource, including its
 2600 ResourceURI and the SelectorSet. This address would be used to retrieve the resource in a subsequent Get
 2601 operation.

2602 The service might use a network address that is the same as the <wsa:To> address in the Create request.

2603 **R7.6-7:** The service may ignore any values in the initial representation that are considered
 2604 read-only from the point of view of the underlying real-world object.

2605 This rule allows Get, Put, and Create to share the same schema. Put also allows the service to ignore
 2606 read-only properties during an update.

2607 **R7.6-8:** If the success of an operation cannot be reported as described in this clause and
 2608 cannot be reversed, the service should return a wsman:EncodingLimit fault with the following
 2609 detail code:

2610 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnreportableSuccess`

2611 7.7 Fragment-Level Access

2612 Because the resource access mechanism defined in this specification works with entire instances and
 2613 it can be inconvenient to specify hundreds or thousands of EPRs just to model fragment-level access
 2614 with full EPRs, WS-Management supports the concept of fragment-level (property) access of
 2615 resources that are normally accessed through the resource access operations. This access is done
 2616 through special use of these operations.

2617 Because of the XML schema limitations discussed in 7.6, simply returning a subset of the XML
 2618 defined for the object being accessed is often incorrect because a subset may violate the XML
 2619 schema for that fragment. To support resource access of fragments or individual elements of a
 2620 representation object, several modifications to the basic resource access operations are made.

2621 **R7.7-1:** A conformant service may support fragment-level access. If the service supports
 2622 fragment-level access, the service shall not behave as if the normal access operations were in
 2623 place but shall operate exclusively on the fragments specified. If the service does not support
 2624 fragment-level access, it shall return a wsman:UnsupportedFeature fault with the following detail
 2625 code:

2626 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FragmentLevelAccess`

2627 **R7.7-2:** A conformant service that supports fragment-level access shall accept the following
 2628 SOAP header in all requests and include it in all responses that transport the fragments:

```

2629 (1) <wsman:FragmentTransfer s:mustUnderstand="true">
2630 (2) xpath to fragment
2631 (3) </wsman:FragmentTransfer>

```

2632 The value of this header is the [XPath 1.0](#) expression that identifies the fragment being transferred
 2633 with relation to the full representation of the object. If an expression other than [XPath 1.0](#) is used,
 2634 a Dialect attribute can be added to indicate this, as follows:

```
2635 (4) <wsman:FragmentTransfer s:mustUnderstand="true"
2636 (5)   Dialect="URIToNewFragmentDialect">
2637 (6)   dialect expression
2638 (7) </wsman:FragmentTransfer>
```

2639 The client needs to understand that unless the header is marked mustUnderstand="true", the service
 2640 might process the request while ignoring the header, resulting in unexpected and potentially serious
 2641 side effects.

2642 XPath is explicitly defined as a dialect due to its importance, but it is not required that
 2643 implementations support XPath as a fragment dialect. Any other type of language to describe
 2644 fragment-level access is permitted as long as the Dialect value is set to indicate to the service what
 2645 dialect is being used.

2646 **R7.7-3:** For resource access fragment operations that use [XPath 1.0](#) (Dialect URI of
 2647 <http://www.w3.org/TR/1999/REC-xpath-19991116>), the value of the
 2648 `/s:Envelope/s:Header/wsman:FragmentTransfer` element is an XPath expression. This XPath
 2649 expression is evaluated using the following context:

- 2650 • **Context Node:** the root element of the XML representation of the resource addressed in
 2651 the request that would be returned as the initial child element of the SOAP Body response if
 2652 a Get operation was applied against the addressed resource without using fragment access
- 2653 • **Context Position:** 1
- 2654 • **Context Size:** 1
- 2655 • **Variable Bindings:** none
- 2656 • **Function Libraries:** Core Function Library [XPath 1.0](#)
- 2657 • **Namespace Declarations:** the [in-scope namespaces] property [XML Infoset](#) of the
 2658 request `/s:Envelope/s:Header/wsman:FragmentTransfer` element

2659 This rule means that the XPath is to be interpreted relative to the XML representation of the resource
 2660 and not relative to any of the SOAP content.

2661 For the Enumeration operations, the XPath is interpreted as defined in clause 8, although the output
 2662 is subsequently wrapped in `wsman:XmlFragment` wrappers after the XPath is evaluated.

2663 An XPath value can refer to the entire node, so the concept of a fragment includes the entire object,
 2664 making fragment-level access a proper superset of normal resource access operations.

2665 If the full XPath expression syntax cannot be supported, a common subset for this purpose is
 2666 described in ANNEX C of this specification. However, in such cases, the Dialect URI is still that of
 2667 XPath.

2668 **R7.7-4:** If a service understands fragment access but does not understand the specified
 2669 fragment Dialect URI or the default dialect, the service shall issue a
 2670 `wsman:FragmentDialectNotSupported` fault.

2671 **R7.7-5:** All resource access messages in either direction of the XML fragments shall be
 2672 wrapped with a `<wsman:XmlFragment>` wrapper that contains a definition that suppresses
 2673 validation and allows any content to pass. A service shall reject any attempt to use
 2674 `wsman:FragmentTransfer` unless the `s:Body` wraps the content using a `wsman:XmlFragment`

2675 wrapper. If any other usage is encountered, the service shall fault the request by using a
2676 wsmt:InvalidRepresentation fault with the following detail code:

2677 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFragment`

2678 Fragment access can occur at any level, including single element, complex elements, simple values,
2679 and attributes. In practice, services typically support only value-level access to elements.

2680 **R7.7-6:** If fragment-level access is supported, a conformant service should support at least
2681 leaf-node, value-level access using an XPath expression that uses the `/text()` NodeTest. In this
2682 case, the value is not wrapped with XML but is transferred directly as text within the
2683 `wsman:XmlFragment` wrapper.

2684 In essence, the transferred content is whatever an XPath operation over the full XML would produce.

2685 **R7.7-7:** If fragment-level access is supported but the filter expression exceeds the capability of
2686 the service, the service should return a `wsman:CannotProcessFilter` fault with text explaining why
2687 the filter was problematic.

2688 **R7.7-8:** For all fragment-level operations, partial successes are not permitted. The entire
2689 meaning of the XPath expression or other dialect shall be fully observed by the service in all
2690 operations, and the entire fragment that is specified shall be successfully transferred in either
2691 direction. Otherwise, faults occur as if none of the operation had succeeded.

2692 All faults are the same as for normal, "full" resource access operations.

2693 The following clauses show how the underlying resource access operations change when transferring
2694 XML fragments.

2695 7.8 Fragment-Level Get

2696 Fragment-level Get is similar to full Get, except for the `wsman:FragmentTransfer` header (lines 25-
2697 27).

2698 EXAMPLE 1: The following example is drawn from the example in 7.1:

```

2699 (1) <s:Envelope
2700 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2701 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2702 (4)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
2703 (5)   <s:Header>
2704 (6)     <wsa:To>
2705 (7)       http://1.2.3.4/wsman
2706 (8)     </wsa:To>
2707 (9)     <wsman:ResourceURI>http://example.org/2005/02/physicalDisk
2708 (10)    </wsman:ResourceURI>
2709 (11)    <wsa:ReplyTo>
2710 (12)    <wsa:Address>
2711 (13)      http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
2712 (14)    </wsa:Address>
2713 (15)  </wsa:ReplyTo>
2714 (16)  <wsa:Action>
2715 (17)    http://schemas.xmlsoap.org/ws/2004/09/transfer/Get
2716 (18)  </wsa:Action>
2717 (19)  <wsa:MessageID>
2718 (20)    urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a87
2719 (21)  </wsa:MessageID>
2720 (22)  <wsman:SelectorSet>
2721 (23)    <wsman:Selector Name="LUN"> 2 </wsman:Selector>

```

```

2722 (23) </wsman:SelectorSet>
2723 (24) <wsman:OperationTimeout> PT30S </wsman:OperationTimeout>
2724 (25) <wsman:FragmentTransfer s:mustUnderstand="true">
2725 (26)     Manufacturer
2726 (27) </wsman:FragmentTransfer>
2727 (28) </s:Header>
2728 (29) <s:Body/>
2729 (30) </s:Envelope>

```

2730 In this case, the service executes the specified XPath expression against the representation that
 2731 would normally have been retrieved, and then return a fragment instead.

2732 **EXAMPLE 2:** The service repeats the wsman:FragmentTransfer element in the GetResponse (lines 48-50) to
 2733 reference the fragment and signal that a fragment has been transferred. The response is wrapped in a
 2734 wsman:XmlFragment wrapper, which suppresses the schema validation that would otherwise apply.

```

2735 (31) <s:Envelope
2736 (32)     xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2737 (33)     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2738 (34)     xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
2739 (35) <s:Header>
2740 (36)     <wsa:To>
2741 (37)         http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
2742 (38)     </wsa:To>
2743 (39)     <wsa:Action s:mustUnderstand="true">
2744 (40)         http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse
2745 (41)     </wsa:Action>
2746 (42)     <wsa:MessageID s:mustUnderstand="true">
2747 (43)         urn:uuid:1a7e7314-d791-4b4b-3eda-c00f7e833a8c
2748 (44)     </wsa:MessageID>
2749 (45)     <wsa:RelatesTo>
2750 (46)         urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a87
2751 (47)     </wsa:RelatesTo>
2752 (48)     <wsman:FragmentTransfer s:mustUnderstand="true">
2753 (49)         Manufacturer
2754 (50)     </wsman:FragmentTransfer>
2755 (51) </s:Header>
2756 (52) <s:Body>
2757 (53)     <wsman:XmlFragment
2758 (54)         xmlns="http://schemas.example.org/2005/02/samples/physDisk">
2759 (54)         <Manufacturer> Acme, Inc. </Manufacturer>
2760 (55)     </wsman:XmlFragment>
2761 (56) </s:Body>
2762 (57) </s:Envelope>

```

2763 The output (lines 53-55) is like that supplied by a typical XPath processor.

2764 To receive the value in isolation without an XML element wrapper, the client can use XPath
 2765 techniques such as the text() operator to retrieve just the values.

2766 **EXAMPLE 3:** The following example request uses text() to get the manufacturer name:

```

2767 (1) <wsman:FragmentTransfer s:mustUnderstand="true">
2768 (2)     Manufacturer/text()
2769 (3) </wsman:FragmentTransfer>

```

2770 This request results in the following XML in the response SOAP Body:

```

2771 (1) <wsman:XmlFragment>
2772 (2)     Acme, Inc.
2773 (3) </wsman:XmlFragment>

```

2774 **7.9 Fragment-Level Put**

2775 Fragment-level Put works like regular Put except that it transfers only the part being updated.
 2776 Although the fragment can be considered part of an instance from the observer's perspective, the
 2777 referenced fragment is treated as the "instance" during the execution of the operation.

2778 NOTE: Put is *always* an update operation of an existing element, whether a simple element or an array. To
 2779 create or insert new elements, Create is required.

2780 EXAMPLE 1: Consider the following XML for illustrative purposes:

```
2781 (1) <a>
2782 (2)   <b>
2783 (3)     <c> </c>
2784 (4)     <d> </d>
2785 (5)   </b>
2786 (6)   <e>
2787 (7)     <f> </f>
2788 (8)     <g> </g>
2789 (9)   </e>
2790 (10) </a>
```

2791 Although <a> is the entire representation of the resource instance, if the operation references the a/b
 2792 node during the Put operation, using an XPath expression of "b", then the content of is updated
 2793 without touching other parts of <a>, such as <e>. If the client wants to update only <d>, then the
 2794 XPath expression used is "b/d".

2795 EXAMPLE 2: Continuing from the example in SECTION 7.1, if the client wanted to update the <BootPartition>
 2796 value from 0 to 1, the following Put fragment could be sent to the service:

```
2797 (1) <s:Envelope
2798 (2)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2799 (3)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2800 (4)   xmlns:wsmn="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
2801 (5)   <s:Header>
2802 (6)     <wsa:To>
2803 (7)       http://1.2.3.4/wsman
2804 (8)     </wsa:To>
2805 (9)     <wsman:ResourceURI>http://example.org/2005/02/physicalDisk
2806 (10)    </wsman:ResourceURI>
2807 (11)    <wsa:ReplyTo>
2808 (12)      <wsa:Address>
2809 (13)        http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
2810 (14)      </wsa:Address>
2811 (15)    </wsa:ReplyTo>
2812 (16)    <wsa:Action>
2813 (17)      http://schemas.xmlsoap.org/ws/2004/09/transfer/Put
2814 (18)    </wsa:Action>
2815 (19)    <wsa:MessageID>
2816 (20)      urn:uuid:d9726315-bc91-2222-9ed8-c044c9658a87
2817 (21)    </wsa:MessageID>
2818 (22)    <wsman:SelectorSet>
2819 (23)      <wsman:Selector Name="LUN"> 2 </wsman:Selector>
2820 (24)    </wsman:SelectorSet>
2821 (25)    <wsman:OperationTimeout> PT30S </wsman:OperationTimeout>
2822 (26)    <wsman:FragmentTransfer s:mustUnderstand="true">
2823 (27)      BootPartition
2824 (28)    </wsman:FragmentTransfer>
2825 (29)  </s:Header>
2826 (30) <s:Body>
```

```

2827 (30) <wsman:XmlFragment>
2828 (31)     <BootPartition> 1 </BootPartition>
2829 (32) </wsman:XmlFragment>
2830 (33) </s:Body>
2831 (34) </s:Envelope>

```

2832 **EXAMPLE 3:** The <BootPartition> wrapper is present because the XPath value specifies this. If
 2833 "BootPartition/text()" were used as the expression, the Body would contain just the value, as in the following
 2834 example:

```

2835 (35) <s:Header>
2836 (36)     ...
2837 (37)     <wsman:FragmentTransfer s:mustUnderstand="true">
2838 (38)         BootPartition/text ()
2839 (39)     </wsman:FragmentTransfer>
2840 (40) </s:Header>
2841 (41) <s:Body>
2842 (42)     <wsman:XmlFragment>
2843 (43)         1
2844 (44)     </wsman:XmlFragment>
2845 (45) </s:Body>

```

2846 If the corresponding update occurs, the new representation matches, so no s:Body result is expected,
 2847 although returning it is always legal. If a value does not match what was requested, the service needs
 2848 to supply only the parts that are different than what is requested. This situation would generally not
 2849 occur for single values because a failure to honor the new value would result in a
 2850 wsmt:InvalidRepresentation fault.

2851 **EXAMPLE 4:** The following is a sample reply:

```

2852 (46) <s:Envelope
2853 (47)     xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2854 (48)     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2855 (49)     xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
2856 (50) <s:Header>
2857 (51)     <wsa:To>
2858 (52)         http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
2859 (53)     </wsa:To>
2860 (54)     <wsa:Action s:mustUnderstand="true">
2861 (55)         http://schemas.xmlsoap.org/ws/2004/09/transfer/PutResponse
2862 (56)     </wsa:Action>
2863 (57)     <wsa:MessageID s:mustUnderstand="true">
2864 (58)         urn:uuid:ee7f13b5-0091-430b-9ed8-2e12fbaa8a7e
2865 (59)     </wsa:MessageID>
2866 (60)     <wsa:RelatesTo>
2867 (61)         urn:uuid:d9726315-bc91-2222-9ed8-c044c9658a87
2868 (62)     </wsa:RelatesTo>
2869 (63)     <wsman:FragmentTransfer s:mustUnderstand="true">
2870 (64)         BootPartition/text ()
2871 (65)     </wsman:FragmentTransfer>
2872 (66) </s:Header>
2873 (67) <s:Body>
2874 (68)     <wsman:XmlFragment>
2875 (69)         1
2876 (70)     </wsman:XmlFragment>
2877 (71) </s:Body>
2878 (72) </s:Envelope>

```

2879 **R7.9-1:** This rule intentionally left blank.

2880 **R7.9-2:** If the service encounters an attempt to update a read-only value using a fragment-level
2881 Put operation, it should return a `wsa:ActionNotSupported` fault with the following detail code:

2882 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ActionMismatch`

2883 NOTE: The fragment-level Put operation implies replacement or update and does not insert new values into the
2884 representation object. Thus, it is not appropriate to use Put to insert a new value at the end of an array, for
2885 example. The entire array can be returned and then updated and replaced (because it is therefore an update of
2886 the entire array), but a single operation to insert a new element in the middle or at the end of an array is actually
2887 a Create operation.

2888 As stated in 7.4, if the new representation differs from the input, the new representation is to be
2889 returned in the response. With fragment-level Put, this rule applies only to the portion of the
2890 representation object being written, not the entire object. If a single value is written and accepted, but
2891 has side effects on other values in the representation, the entire object is *not* returned.

2892 To set a value to NULL without removing it as an element, use an attribute value of `xsi:nil` on the
2893 element being set to NULL to ensure that the fragment path is adjusted appropriately.

2894 EXAMPLE 5:

```
2895 (73) <s:Header> ...  
2896 (74) <wsman:FragmentTransfer s:mustUnderstand="true">  
2897 (75)   AssetLabel  
2898 (76) </wsman:FragmentTransfer>  
2899 (77)   ...  
2900 (78) </Header>  
2901 (79) <s:Body>  
2902 (80)   <wsman:XmlFragment xmlns:xsi="www.w3.org/2001/XMLSchema-instance">  
2903 (81)     <AssetLabel xsi:nil="true"/>  
2904 (82)   </wsman:XmlFragment>  
2905 (83) </s:Body>
```

2906 7.10 Fragment-Level Delete

2907 Fragment-level Delete applies only if the XML schema for the targeted object supports optional
2908 elements that can be removed from the representation object, or supports arrays (repeated elements)
2909 with varying numbers of elements and the client wants to remove an element in an array. If
2910 replacement of an entire array is needed, fragment-level Put can be used. For array access, the
2911 XPath array access notation can conveniently be used. To delete a value that is legal to remove
2912 (according to the rules of the schema for the object), the `wsman:FragmentTransfer` expression
2913 identifies the item to be removed.

2914 EXAMPLE 1:

```
2915 (1) <wsman:FragmentTransfer s:mustUnderstand="true">  
2916 (2)   VolumeLabel  
2917 (3) </wsman:FragmentTransfer>
```

2918 To set a value to NULL without removing it as an element, use fragment-level Put with a value of
2919 `xsi:nil`.

2920 To delete an array element, use the XPath [] operators.

2921 EXAMPLE 2: The following example deletes the second <BlockedIPAddress> element in the representation.
2922 (XPath arrays are 1 based.)

```
2923 (1) <wsman:FragmentTransfer s:mustUnderstand="true">
2924 (2)   BlockedIPAddress[2]
2925 (3) </wsman:FragmentTransfer>
```

2926 The <s:Body> is empty for all Delete operations, even with fragment-level access, and all normal
2927 faults for Delete apply.

2928 **R7.10-1:** If a value cannot be deleted because of locking conditions or similar phenomena, the
2929 service should return a wsman:AccessDenied fault.

2930 7.11 Fragment-Level Create

2931 Fragment-level Create applies only if the XML schema for the targeted object supports optional
2932 elements that are not currently present, or supports arrays with varying numbers of elements and the
2933 client wants to insert an element in an array (a repeated element). If entire array replacement is
2934 needed, Fragment-level Put can be used. For array access, the XPath array access notation (the []
2935 operators) can be used.

2936 NOTE: Create can be used only to add new content, not to update existing content.

2937 To insert a value that can be legally added (according to the rules of the schema for the object), the
2938 wsman:FragmentTransfer expression identifies the item to be added.

2939 EXAMPLE 1: For example, assume the following message fragment is sent to a LogicalDisk resource:

```
2940 (1) <wsman:FragmentTransfer s:mustUnderstand="true">
2941 (2)   VolumeLabel
2942 (3) </wsman:FragmentTransfer>
```

2943 EXAMPLE 2: In this case, the <Body> contains both the element and the value:

```
2944 (4) <s:Body>
2945 (5)   <wsman:XmlFragment>
2946 (6)     <VolumeLabel> MyDisk </VolumeLabel>
2947 (7)   </wsman:XmlFragment>
2948 (8) </s:Body>
```

2949 This operation creates a <VolumeLabel> element where none existed before.

2950 EXAMPLE 3: To create the target using the value alone, apply the XPath text() operator to the path, as follows:

```
2951 (9) <wsman:FragmentTransfer s:mustUnderstand="true">
2952 (10)   VolumeLabel/text()
2953 (11) </wsman:FragmentTransfer>
```

2954 EXAMPLE 4: The body of Create contains the value to be inserted and is the same as for fragment-level Put:

```
2955 (12) <s:Body>
2956 (13)   <wsman:XmlFragment>
2957 (14)     MyDisk
2958 (15)   </wsman:XmlFragment>
2959 (16) </s:Body>
```

2960 To create an array element in the target, the XPath [] operator may be used. To insert a new element
2961 at the end of the array, the user needs to know the number of elements in the array so that the new
2962 index can be used.

2963 EXAMPLE 5: The following message fragment is sent to an InternetServer resource:

```

2964 (17) <wsman:FragmentTransfer s:mustUnderstand="true">
2965 (18)   BlockedIPAddress[3]
2966 (19) </wsman:FragmentTransfer>

```

2967 Insertion of a new element within the array is done using the index of the desired location, and the
 2968 array expands at that location to accommodate the new element. Using Put at this location *overwrites*
 2969 the existing array element, whereas Create inserts a *new* element, making the array larger.

2970 The body of Create contains the value to be inserted and is the same as for fragment-level Put.

2971 EXAMPLE 6:

```

2972 (20) <s:Body>
2973 (21)   <wsman:XmlFragment>
2974 (22)     <BlockedIPAddress> 123.12.188.44 </BlockedIPAddress>
2975 (23)   </wsman:XmlFragment>
2976 (24) </s:Body>

```

2977 This operation adds a third IP address to the <BlockedIPAddress> array (a repeated element),
 2978 assuming that at least two elements are at that level already.

2979 **R7.11-1:** A service shall not use fragment-level Create to modify the value of an existing
 2980 property. If the targeted object and the targeted property already exists, the service should return
 2981 a wsman:AlreadyExists fault.

2982 **R7.11-2:** If the Create fails because the result would not conform to the schema in some way,
 2983 the service should return a wsmt:InvalidRepresentation fault.

2984 As defined in 7.6, the CreateResponse contains the EPR of the created resource. In the case of
 2985 fragment-level Create, the response additionally contains the wsman:FragmentTransfer block,
 2986 including the path (line 12), in a SOAP header.

2987 EXAMPLE 7: In the following example, the ResourceCreated EPR continues to refer to the entire object, not just
 2988 to the fragment.

```

2989 (25) <s:Envelope
2990 (26)   xmlns:s="http://www.w3.org/2003/05/soap-envelope"
2991 (27)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
2992 (28)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
2993 (29)   xmlns:wsmt="http://schemas.xmlsoap.org/ws/2004/09/transfer">
2994 (30) <s:Header>
2995 (31)   ...
2996 (32)   <wsa:Action>
2997 (33)     http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse
2998 (34)   </wsa:Action>
2999 (35) <wsman:FragmentTransfer s:mustUnderstand="true">
3000 (36)   Path To Fragment
3001 (37) </wsman:FragmentTransfer>
3002 (38)   ...
3003 (39) </s:Header>
3004 (40) <s:Body>
3005 (41)   <wsmt:ResourceCreated>
3006 (42)     <wsa:Address> ... </wsa:Address>
3007 (43)     <wsa:ReferenceParameters>
3008 (44)       <wsman:SelectorSet>
3009 (45)         <wsman:Selector ...> ... </wsman:Selector>
3010 (46)       </wsman:SelectorSet>
3011 (47)     </wsa:ReferenceParameters>
3012 (48)   </wsmt:ResourceCreated>
3013 (49) </s:Body>

```

3014 (50) </s:Envelope>

3015 As discussed in 7.6, to remain compatible with WSDL, only the EPR of the item is returned in the
3016 SOAP Body, in spite of other options discussed in 7.6.

3017 **8 Enumeration of Datasets**

3018 **8.1 General**

3019 This clause defines a set of operations that can be used as a basis for iteration through the members
3020 of a management-specific dataset or collection. WS-Management qualifies and extends these
3021 operations as described in this clause.

3022 There are numerous applications for which a simple single-request/single-reply metaphor is
3023 insufficient for transferring large data sets over SOAP. Applications that do not fit into this simple
3024 paradigm include streaming, traversal, query, and enumeration.

3025 This clause defines a simple SOAP-based protocol for enumeration that allows the data source to
3026 provide a session abstraction, called an enumeration context, to a consumer that represents a logical
3027 cursor through a sequence of data items. The consumer can then request XML element information
3028 items using this enumeration context over the span of one or more SOAP messages.

3029 Somewhere, state must be maintained regarding the progress of the iteration. This state may be
3030 maintained between requests by the data source being enumerated or by the data consumer. The
3031 operations defined in this clause allow the data source to decide, on a request-by-request basis,
3032 which party is responsible for maintaining this state for the next request.

3033 In its simplest form, there is a single operation, Pull, which allows a data source, in the context of a
3034 specific enumeration, to produce a sequence of XML elements in the body of a SOAP message.
3035 Each subsequent Pull operation returns the next N elements in the aggregate sequence.

3036 A data source may provide a custom mechanism for starting a new enumeration. For instance, a data
3037 source that provides access to a SQL database may support a SELECT operation that performs a
3038 database query and uses an explicit database cursor to iterate through the returned rows. In general,
3039 however, it is simpler if all data sources support a single, standard operation to start an enumeration.
3040 This specification defines such an operation, Enumerate, which data sources may implement for
3041 starting a new enumeration of a data source. The Enumerate operation is used to create new
3042 enumeration contexts for subsequent traversal/retrieval. Each Enumerate operation results in a
3043 distinct enumeration context, each with its own logical cursor/position.

3044 It should be emphasized that different enumerations of the same data source may produce different
3045 results; this may happen even for two enumeration contexts created concurrently by a single
3046 consumer using identical Enumerate requests. In general, the consumer of an enumeration should
3047 not make any assumptions about the ordering or completeness of the enumeration; the returned data
3048 items represent a selection by the data source of items it wishes to present to that consumer at that
3049 time in that order, with no guarantee that every available item is returned or that the order in which
3050 items is returned has any semantic meaning whatsoever (of course, any specific data source may
3051 provide strong guarantees, if so desired). In particular, it should be noted that the very act of
3052 enumerating the contents of a data source may modify the contents of the data source; for instance, a
3053 queue might be represented as a data source such that items that are returned in a Pull response are
3054 removed from the queue.

3055 Enumeration contexts represent a specific traversal through a sequence of XML information items. An
3056 Enumerate operation may be used to establish an enumeration context from a data source. A Pull
3057 operation is used to fetch information items from a data source according to a specific enumeration
3058 context. A Release operation is used to tell a data source that the consumer is abandoning an
3059 enumeration context before it has completed the enumeration.

3060 Enumeration contexts are represented as XML data that is opaque to the consumer. Initially, the
3061 consumer gets an enumeration context from the data source by means of an Enumerate operation.
3062 The consumer then passes that XML data back to the data source in the Pull request. Optionally, the
3063 data source may return an updated enumeration context in the Pull response; when present, this new
3064 enumeration context should replace the old one on the consumer, and it should be passed to the data
3065 source in all future responses until and unless the data source again returns an updated enumeration
3066 context.

3067 Consumers should not reuse old enumeration contexts that have been replaced by the data source.
3068 Using a replaced enumeration context in a Pull response may yield undefined results, including being
3069 ignored or generating a fault.

3070 After the last element in a sequence has been returned, or the enumeration context has expired, the
3071 enumeration context is considered invalid and the result of subsequent operations referencing that
3072 context is undefined.

3073 Callers may issue a Release operation against a valid enumeration context at any time, which causes
3074 the enumeration context to become invalid and allows the data source to free up any resources it may
3075 have allocated to the enumeration. Issuing a Release operation prior to reaching the end of the
3076 sequence of elements is explicitly allowed; however, no further operations should be issued after a
3077 Release.

3078 In addition, the data source may invalidate an enumeration context at any time, as necessary.

3079 If a resource with multiple instances provides a mechanism for enumerating or querying the set of
3080 instances, the operations defined in this clause can be used to perform the iteration.

3081 **R8.1-1:** A service may support the Enumeration operations if enumeration of any kind is
3082 supported.

3083 **R8.1-2:** If simple, unfiltered enumeration of resource instances is exposed through Web
3084 services, a conformant service shall support the Enumeration operations to expose this. The
3085 service may also support other techniques for enumerating the instances.

3086 **R8.1-3:** If filtered enumeration (queries) of resource instances is exposed through Web
3087 services, a conformant service should support the Enumeration operations to expose this. The
3088 service may also support other techniques for enumerating the instances.

3089 This clause indicates that enumeration is a three-part operation:

3090 1) An initial Enumerate message is issued to establish the enumeration context.
3091 2) Pull operations are used to iterate over the result set.
3092 3) When the enumeration iterator is no longer required and not yet exhausted, a Release
3093 message is issued to release the enumerator and associated resources.

3094 As with other WS-Management methods, the enumeration can make use of wsman:OptionSet.

3095 **R8.1-4:** A service may implement wsmen:Renew, wsmen:GetStatus and
3096 wsmen:EnumerationEnd messages; however, in constrained environments these are candidates
3097 for exclusion. If these messages are not supported, then a wsa:ActionNotSupported fault shall be
3098 returned in response to these requests.

3099 **R8.1-5:** If a service is exposing enumeration, it shall at least support the following messages:
3100 Enumerate, Pull, and Release, and their associated responses.

3101 If the service does not support stateful enumerators, the Release is a simple no-op, so it is trivial to
 3102 implement. (It always succeeds when the operation is valid.) However, it is supported to allow for the
 3103 uniform construction of clients.

3104 **R8.1-6:** The Pull and Release operations are a continuation of the original Enumerate
 3105 operation. The service should enforce the same authentication and authorization throughout the
 3106 entire sequence of operations and should fault any attempt to change credentials during the
 3107 sequence.

3108 Some transports such as HTTP might drop or reestablish connections between Enumerate and
 3109 subsequent Pull operations, or between Pull operations. It is expected that services will allow the
 3110 enumeration to continue uninterrupted, but for practical reasons some services might require that the
 3111 same connection be used. This specification establishes no requirements in this regard. However,
 3112 R8.1-6 establishes that the user credentials do not change during the entire enumeration sequence.

3113 8.2 Enumerate

3114 All data sources shall support some operation that allows an enumeration to be started. A data
 3115 source may support the Enumerate operation, or it may provide some other mechanism for starting
 3116 an enumeration and receiving an enumeration context.

3117 The Enumerate operation is initiated by sending an Enumerate request message to the data source.
 3118 The Enumerate request message shall be of the following form:

```

3119 (1) <s:Envelope ...>
3120 (2)   <s:Header ...>
3121 (3)     <wsa:Action>
3122 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate
3123 (5)     </wsa:Action>
3124 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
3125 (7)     <wsa:To>xs:anyURI</wsa:To>
3126 (8)     ...
3127 (9)   </s:Header>
3128 (10)  <s:Body ...>
3129 (11)   <wsmen:Enumerate ...>
3130 (12)     <wsmen:EndTo>endpoint-reference</wsmen:EndTo> ?
3131 (13)     <wsmen:Expires>[xs:dateTime | xs:duration]</wsmen:Expires> ?
3132 (14)     <wsmen:Filter Dialect="xs:anyURI"?> xs:any </wsmen:Filter> ?
3133 (15)     ...
3134 (16)   </wsmen:Enumerate>
3135 (17)  </s:Body>
3136 (18) </s:Envelope>
  
```

3137 The following describes additional, normative constraints on the preceding outline:

3138 /s:Envelope/s:Header/wsa:Action

3139 This required element shall contain the value:

3140 `http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate.`

3141 If a SOAP Action URI is also present in the underlying transport, its value shall convey the same
 3142 value.

3143 /s:Envelope/s:Body/*/wsmen:EndTo

3144 This optional element denotes where to send an EnumerationEnd message if the enumeration is
 3145 terminated unexpectedly. If present, this element shall be of type `wsa:EndpointReferenceType`.
 3146 The default is to not send this message. The endpoint referenced by this EPR shall implement a
 3147 binding of the "EnumEndEndpoint" portType described in ANNEX H.

- 3148 /s:Envelope/s:Body*/wsmen:Expires
- 3149 Requested expiration time for the enumeration. (No implied value.) The data source defines the
- 3150 actual expiration and is not constrained to use a time less or greater than the requested
- 3151 expiration. The expiration time may be a specific time or a duration from the enumeration's
- 3152 creation time. Both specific times and durations are interpreted based on the data source's clock.
- 3153 If this element does not appear, then the request is for an enumeration that will not expire. That
- 3154 is, the consumer is requesting the data source to create an enumeration with an indefinite
- 3155 lifetime. If the data source grants such an enumeration, it will terminate when the end of the
- 3156 enumeration is reached, or if the consumer sends a Release request, or by the data source at
- 3157 any time for reasons such as connection termination, resource constraints, or system shut-down.
- 3158 If the expiration time is either a zero duration or a specific time that occurs in the past according
- 3159 to the data source, then the request shall fail, and the data source may generate a
- 3160 wsmen:InvalidExpirationTime fault indicating that an invalid expiration time was requested.
- 3161 Some data sources may not have a "wall time" clock available, and so are able only to accept
- 3162 durations as expirations. If such a source receives an Enumerate request containing a specific
- 3163 time expiration, then the request shall fail; if so, the data source should generate a
- 3164 wsmen:UnsupportedExpirationType fault indicating that an unsupported expiration type was
- 3165 requested.
- 3166 /s:Envelope/s:Body/wsmen:Enumerate/wsmen:Filter
- 3167 This optional element contains a Boolean predicate in some dialect (see
- 3168 /s:Envelope/s:Body*/wsmen:Filter/@Dialect) that all elements of interest must satisfy. The
- 3169 resultant enumeration context shall not return elements for which this predicate expression
- 3170 evaluates to the value false. If this element is absent, then the implied value is the expression
- 3171 true(), indicating that no filtering is desired.
- 3172 If the data source does not support filtering, the request shall fail, and the data source may
- 3173 generate a wsmen:FilteringNotSupported SOAP fault as follows:
- 3174 If the data source supports filtering but cannot honor the requested filter dialect, the request shall
- 3175 fail, and the data source may generate a wsmen:FilterDialectRequestedUnavailable SOAP fault
- 3176 as follows:
- 3177 If the data source supports filtering and the requested dialect but cannot process the requested
- 3178 filter content, the request shall fail, and the data source may generate a
- 3179 wsman:CannotProcessFilter SOAP fault as follows:
- 3180 /s:Envelope/s:Body*/wsmen:Filter/@Dialect
- 3181 Implied value is "http://www.w3.org/TR/1999/REC-xpath-19991116".
- 3182 /s:Envelope/ s:Body/ */ wsmen:Filter/ @Dialect= "http://www.w3.org/TR/1999/REC-xpath-19991116"
- 3183 Value of /s:Envelope/s:Body*/wsmen:Filter is an XPath [XPath 1.0] predicate expression
- 3184 (PredicateExpr); the context of the expression is:
- 3185 • **Context Node:** any XML element that could be returned as a direct child of the Items
 - 3186 element
 - 3187 • **Context Position:** 1
 - 3188 • **Context Size:** 1
 - 3189 • **Variable Bindings:** None
 - 3190 • **Function Libraries:** Core Function Library [XPath 1.0]
 - 3191 • **Namespace Declarations:** The [in-scope namespaces] property [[XML Infoset](#)] of
 - 3192 /s:Envelope/s:Body*/wsmen:Filter
- 3193 Other components of the preceding outline are not further constrained by this specification.

3194 Upon successful processing of an Enumerate request message, a data source is expected to create
 3195 an enumeration context and return that context in an Enumerate response message, which shall
 3196 adhere to the following form:

```

3197 (1) <s:Envelope ...>
3198 (2)   <s:Header ...>
3199 (3)     <wsa:Action>
3200 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerateResponse
3201 (5)     </wsa:Action>
3202 (6)     <wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
3203 (7)     <wsa:To>xs:anyURI</wsa:To>
3204 (8)     ...
3205 (9)   </s:Header>
3206 (10)  <s:Body ...>
3207 (11)   <wsmen:EnumerateResponse ...>
3208 (12)     <wsmen:Expires>[xs:dateTime | xs:duration]</wsmen:Expires> ?
3209 (13)     <wsmen:EnumerationContext>...</wsmen:EnumerationContext>
3210 (14)     ...
3211 (15)   </wsmen:EnumerateResponse>
3212 (16)  </s:Body>
3213 (17) </s:Envelope>

```

3214 The following describes additional, normative constraints on the preceding outline:

3215 /s:Envelope/s:Header/wsa:Action

3216 This required element shall contain the value:

3217 `http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerateResponse`

3218 If a SOAP Action URI is also present in the underlying transport, its value shall convey the same
 3219 value.

3220 /s:Envelope/s:Body/*/wsmen:Expires

3221 The expiration time assigned by the data source. The expiration time may be either an absolute
 3222 time or a duration but should be of the same type as the requested expiration (if any).

3223 If this element does not appear, then the enumeration will not expire. That is, the enumeration
 3224 has an indefinite lifetime. It will terminate when the end of the enumeration is reached, if the
 3225 consumer sends a Release request, or by the data system at any time for reasons such as
 3226 connection termination, resource constraints, or system shut-down.

3227 /s:Envelope/s:Body/wsmen:EnumerateResponse/wsmen:EnumerationContext

3228 The required EnumerationContext element contains the XML representation of the new
 3229 enumeration context. The consumer is required to pass this XML data in Pull requests for this
 3230 enumeration context, until and unless a PullResponse message updates the enumeration
 3231 context.

3232 8.2.1 General

3233 WS-Management qualifies the Enumerate operation as described in this clause.

3234 **R8.2.1-1:** A conformant service may accept a wsmen:Enumerate message with an EndTo
 3235 address; however, if EnumerationEnd is not supported, a service may instead issue a
 3236 wsman:UnsupportedFeature fault with the following detail code:

3237 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode`

3238 **R8.2.1-2:** A conformant service shall accept an Enumerate message with an Expires timeout
 3239 or fault with wsman:UnsupportedFeature and the following detail code:

3240 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ExpirationTime`

3241 **R8.2.1-3:** The wsman:Filter element (see 8.3) in the Enumerate body shall be either simple
 3242 text or a single complex XML element. A conformant service shall not accept mixed content of
 3243 both text and elements, or multiple peer XML elements under the wsman:Filter element.

3244 Although this use of mixed content is allowed in the general case of Enumerate, it is unnecessarily
 3245 complex for WS-Management implementations.

3246 A common filter dialect is [XPath 1.0](http://www.w3.org/TR/1999/REC-xpath-19991116) (identified by the Dialect URI <http://www.w3.org/TR/1999/REC-xpath-19991116>). Resource-constrained implementations might have difficulty exporting full XPath
 3247 processing and yet still want to use a subset of XPath syntax. As long as the filter expression is a
 3248 proper subset of the specified dialect, it is legal and can be described using that Dialect value.
 3249

3250 No rule mandates the use of XPath or any subset as a filtering dialect. If no Dialect is specified, the
 3251 default interpretation is that the Filter value is XPath (as specified previously in this clause).

3252 **R8.2.1-4:** A conformant service may not support the entire syntax and processing power of
 3253 the specified Filter Dialect. The only requirement is that the specified Filter is syntactically correct
 3254 within the definition of the Dialect. Subsets are therefore legal. If the specified Filter exceeds the
 3255 capability of the service, the service should return a wsman:CannotProcessFilter fault with some
 3256 text indicating what went wrong.

3257 Some services require filters to function because their search space is so large that simple
 3258 enumeration is meaningless or impossible.

3259 **R8.2.1-5:** If a wsman:Filter is required, a conformant service shall fault any request without a
 3260 wsman:Filter, by using a wsman:UnsupportedFeature fault with the following detail code:

3261 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FilteringRequired`

3262 **R8.2.1-6:** A conformant service may block, fault (using wsman:Concurrency faults), or allow
 3263 other concurrent operations on the resource for the duration of the enumeration, and may include
 3264 or exclude the results of such operations as part of any enumeration still in progress.

3265 If clients execute other operations, such as Create or Delete, while an enumeration is occurring, this
 3266 specification makes no restrictions on the behavior of the enumeration. The service can include or
 3267 exclude the results of these operations in real-time, can produce an initial snapshot of the
 3268 enumeration and execute the Pull requests from this snapshot, or can deny access to other
 3269 operations while enumerations are in progress.

3270 8.2.2 Enumeration "Count" Option

3271 To give clients an estimate of the number of items in an enumeration, two optional SOAP headers are
 3272 defined: one for use in the request message to return an approximate count of items in an
 3273 enumeration sequence, and a corresponding header for use in the response to return this value to the
 3274 client.

3275 These SOAP headers are defined for use with the Enumerate and Pull messages and their
 3276 responses. The header used in Enumerate and Pull is as follows:

```
3277 (1) <s:Header>
3278 (2)   ...
3279 (3)   <wsman:RequestTotalItemsCountEstimate .../>
3280 (4) </s:Header>
```

3281 The header used by the service to return the value is as follows:

```
3282 (5) <s:Header>
3283 (6)   ...
3284 (7)   <wsman:TotalItemsCountEstimate>
```

```

3285 (8) xs:nonNegativeInteger
3286 (9) </wsman:TotalItemsCountEstimate>
3287 (10) </s:Header>

```

3288 The following definitions provide additional, normative constraints on the preceding headers:

3289 wsman:RequestTotalItemsCountEstimate

3290 when present as a SOAP header on an Enumerate or Pull message, indicates that the client is
 3291 requesting that the associated response message includes an estimate of the total number of
 3292 items in the enumeration sequence

3293 This SOAP header does not have any meaning defined by this specification when included with
 3294 any other messages.

3295 wsman:TotalItemsCountEstimate

3296 when present as a SOAP header on an EnumerateResponse or PullResponse message,
 3297 indicates the approximate number of items in the enumeration sequence

3298 This is the total number of items and not the remaining number of items in the sequence. This
 3299 SOAP header does not have any meaning defined by this specification when included with any
 3300 other messages.

3301 When a service understands the TotalItemsCountEstimate feature but cannot determine the
 3302 number of items, the service responds with the wsman:TotalItemsCountEstimate element having
 3303 an xsi:nil attribute with value 'true', and having no value, as follows:

```

3304 (1) <wsman:TotalItemsCountEstimate xsi:nil="true"/>

```

3305 **R8.2.2-1:** A conformant service may support the ability to return an estimate of the number of
 3306 items in an enumeration sequence. If a service receives an Enumerate or Pull message without
 3307 the wsman:RequestTotalItemsCountEstimate SOAP header, the service shall not return the
 3308 wsman:TotalItemsCountEstimate SOAP header on the associated response message.

3309 **R8.2.2-2:** The value returned in the wsman:TotalItemsCountEstimate SOAP header is only an
 3310 estimate of the number of items in the sequence. The client should not use the
 3311 wsman:TotalItemsCountEstimate value for determining an end of enumeration instead of using
 3312 EndOfSequence.

3313 This mechanism is intended to assist clients in determining the percentage of completion of an
 3314 enumeration as it progresses. When a service sends a result count estimate after a previous estimate
 3315 for the same enumeration sequence, the most recent total results count estimate is considered to be
 3316 the more precise estimate.

3317 8.2.3 Optimization for Enumerations with Small Result Sets

3318 To optimize the number of round-trip messages required to enumerate the items in an enumerable
 3319 resource, a client can request optimized enumeration behavior. This behavior is useful in cases
 3320 where the enumeration has such a small number of items that the initial EnumerateResponse could
 3321 reasonably include the entire result, without the need for a subsequent Pull to retrieve the items. This
 3322 mechanism can be used even for large enumerations to get the first few results in the initial response.

3323 A client initiates an optimized enumeration by placing the wsman:OptimizeEnumeration element as a
 3324 child element of the Enumerate element, and can optionally include the wsman:MaxElements
 3325 element, as follows:

3326 EXAMPLE:

```

3327 (1) <s:Body>

```

```

3328 (2) <wsmen:Enumerate>
3329 (3)   ...
3330 (4)   <wsman:OptimizeEnumeration/>
3331 (5)   <wsman:MaxElements>xs:positiveInteger</wsman:MaxElements> ?
3332 (6)   </wsmen:Enumerate>
3333 (7) </s:Body>

```

3334 The following definitions provide additional, normative constraints on the preceding outline:

3335 `wsmen:Enumerate/wsman:OptimizeEnumeration`

3336 when present as a child of the `Enumerate` element, indicates that the client is requesting an
 3337 optimized enumeration

3338 `wsmen:Enumerate/wsman:MaxElements`

3339 (optional) indicates the maximum number of items the consumer is willing to accept in the
 3340 `EnumerateResponse`

3341 It plays the same role as `wsmen:Pull/wsmen:MaxElements`. When this element is absent, its
 3342 implied value is 1.

3343 **R8.2.3-1:** A conformant service may support enumeration optimization. If a service receives
 3344 the `wsman:OptimizeEnumeration` element in an `Enumerate` message and it does not support
 3345 enumeration optimization, it should ignore the element and complete the enumeration request as
 3346 if the element were not present.

3347 If the service ignores the element, the client continues with a subsequent `Pull` as if the option was not
 3348 in force. The client requires no special mechanisms over what was needed for normal enumeration if
 3349 the optimization request is ignored.

3350 **R8.2.3-2:** A conformant service that receives an `Enumerate` message without the
 3351 `wsman:OptimizeEnumeration` element shall not return any enumeration items in the
 3352 `EnumerateResponse` message and shall return a `EnumerationContext` initialized to return the first
 3353 items when the first `Pull` message is received.

3354 If the service implements the optimization even if it was not requested, clients unaware of the
 3355 optimization will incorrectly process the enumeration result.

3356 **R8.2.3-3:** A conformant service that receives an `Enumerate` message with the
 3357 `wsman:OptimizeEnumeration` element shall not return more elements in the `Enumerate` response
 3358 message than requested in the `wsman:MaxElements` element (or no more than 1 item if the
 3359 `wsman:MaxElements` element is not present). Implementations may return fewer items based on
 3360 either the `wsman:OperationTimeout` SOAP header, `wsman:MaxEnvelopeSize` SOAP header, or
 3361 implementation-specific constraints.

3362 When requested by the client, a service implementing the optimized enumeration will respond with
 3363 the following additional content in an `EnumerateResponse` message:

```

3364 (1) <s:Body>
3365 (2)   <wsmen:EnumerateResponse>
3366 (3)     <wsmen:EnumerationContext> ... </wsmen:EnumerationContext>
3367 (4)     <wsman:Items>
3368 (5)       ...same as for wsman:Items in wsman:PullResponse
3369 (6)     </wsman:Items> ?
3370 (7)     <wsman:EndOfSequence/> ?
3371 (8)     ...
3372 (9)   </wsmen:EnumerateResponse>
3373 (10) </s:Body>

```

3374 The following definitions provide additional, normative constraints on the preceding outline:

3375 wsman:Items

3376 (optional) contains one or more enumeration-specific elements as would have been encoded for
3377 Items in a PullResponse

3378 The service will return no more than wsman:MaxElements elements in this list if
3379 wsman:MaxElements is specified in the request message, or one element if
3380 wsman:MaxElements was omitted.

3381 wsman:EndOfSequence

3382 (optional) indicates that no more elements are available from this enumeration and that the
3383 entire result (even if there are zero elements) is contained within the wsman:Items element

3384 wsmen:EnumerationContext

3385 required context for requesting additional items, if any, in subsequent Pull messages

3386 If the wsman:EndOfSequence is also present, the EnumerationContext cannot be used in a
3387 subsequent Pull request. The service should observe the same fault usage that would occur if
3388 the EnumerationContext were used in a Pull request after the EndOfSequence element occurred
3389 in a PullResponse. Although the EnumerationContext element must be present, no value is
3390 required; therefore, in cases where the wsman:EndOfSequence element is present, the value for
3391 EnumerationContext can be empty.

3392 EXAMPLE:

```
3393 (1) <s:Body>
3394 (2)   <wsmen:EnumerateResponse>
3395 (3)     <wsmen:EnumerationContext/>
3396 (4)     <wsman:Items>
3397 (5)       Items
3398 (6)     </wsman:Items>
3399 (7)     <wsman:EndOfSequence/>
3400 (8)     ...
3401 (9)   </wsmen:EnumerateResponse>
3402 (10) </s:Body>
```

3403 **R8.2.3-4:** A conformant service that supports optimized enumeration and is responding with
3404 an EnumerateResponse message shall include the wsman:Items element, the
3405 wsman:EndOfSequence element, or both in the response as an indication to the client that the
3406 optimized enumeration request was understood and honored.

3407 If neither wsman:Items nor wsman:EndOfSequence is in the EnumerateResponse message, the
3408 client can continue to use the enumeration message exchanges as defined in 8.2.1.

3409 **R8.2.3-5:** A conformant service that supports optimized enumeration and has not returned all
3410 items of the enumeration sequence in the EnumerateResponse message shall return an
3411 EnumerationContext element that is initialized such that a subsequent Pull message will return
3412 the set of items after those returned in the EnumerateResponse. If all items of the enumeration
3413 sequence have been returned in the EnumerateResponse message, the service should return an
3414 empty EnumerationContext element and shall return the wsman:EndOfSequence element in the
3415 response.

3416 A client that has requested optimized enumeration can determine if this request was understood and
3417 honored by the service by examining the response message.

3418 Clients concerned about the size of the initial response, irrespective of the number of items, can use
3419 the wsman:MaxEnvelopeSize mechanism described in 6.2.

3420 8.3 Filter Interpretation

3421 The Filter expression is constrained to be a Boolean predicate. To support ad hoc queries including
3422 projections, WS-Management defines a wsman:Filter element of exactly the same form as in the
3423 Enumeration filter except that the filter expression is not constrained to be a Boolean predicate. This
3424 allows the use of enumeration using existing query languages such as SQL and CQL, which combine
3425 predicate and projection information in the same syntax. The use of projections is defined by the filter
3426 dialect, not by WS-Management.

3427 `(1) <wsman:Filter Dialect="xs:anyURI"?> xs:any </wsman:Filter>`

3428 The Dialect attribute is optional. When not specified, it has the following implied value:

3429 `http://www.w3.org/TR/1999/REC-xpath-19991116`

3430 This dialect allows any full XPath expression or subset to be used.

3431 The wsman:Filter element is a child of the Enumerate element.

3432 If the filter dialect used for the Enumerate message is XPath 1.0, the context node is the same as that
3433 specified in 8.1.

3434 **R8.3-1:** If a service supports filtered enumeration using Filter, it shall also support filtering using
3435 wsman:Filter. This rule allows client stacks to always pick the wsman XML namespace for the
3436 Filter element. Even though a service supports wsman:Filter, it is not required to support
3437 projections.

3438 **R8.3-2:** If a service supports filtered enumeration using wsman:Filter, it should also support
3439 filtering using Filter.

3440 **R8.3-3:** If an Enumerate request contains both Filter and wsman:Filter, the service shall return
3441 a wsman:CannotProcessFilter fault.

3442 Filters are generally intended to select entire XML document representations. However, most query
3443 languages have both filtering and compositional capabilities in that they can return subsets of the
3444 original representation, or perform complex operations on the original representation and return
3445 something entirely new.

3446 This specification places no restriction on the capabilities of the service, but services may elect to
3447 provide only simple filtering capability and no compositional capabilities. In general, filtering dialects
3448 fall into the following simple hierarchy:

- 3449 1) simple enumeration with no filtering
- 3450 2) filtered enumeration with no representation change (within the capabilities of XPath, for
3451 example)
- 3452 3) filtered enumeration in which a subset of each item is selected (within the capabilities of
3453 XPath, for example)
- 3454 4) composition of new output (XQuery), including simple projection

3455 Most services fall into the first or second category. However, if a service wants to support fragment-
3456 level enumeration to complement fragment-level access (7.7), the service can implement category 3
3457 as well. Only rarely do services implement category 4.

3458 [XPath 1.0](#) can be used simply for filtering, or it can be used to send back subsets of the
3459 representation (or even the values without XML wrappers). In cases where the result is not just
3460 filtered but also "altered," the technique in 8.6 applies.

3461 If full XPath cannot be supported, a common subset for this purpose is described in D.3 of this
3462 specification.

3463 EXAMPLE 1: Following is a typical example of the use of XPath in a filter. Assume that each item in the
3464 enumeration to be delivered has the following XML content:

```
3465 (1) <s:Body>
3466 (2)   ...
3467 (3)   <wsmen:Items>
3468 (4)     <DiskInfo xmlns="...">
3469 (5)       <LogicalDisk>C:</LogicalDisk>
3470 (6)       <CurrentMegabytes>12</CurrentMegabytes>
3471 (7)       <BackupDrive> true </BackupDrive>
3472 (8)     </DiskInfo>
3473 (9)     ...
3474 (10)  </wsmen:Items>
3475 (11) </s:Body>
```

3476 The anchor point for the XPath evaluation is at the first element of each item within the Items
3477 wrapper, and it does not reference the s:Body or Items elements. The XPath expression is evaluated
3478 as if each item in the Items block were a separate document.

3479 EXAMPLE 2: When used for simple document processing, the following four XPath expressions "select" the
3480 entire DiskInfo node:

```
3481 (12) /
3482 (13) /DiskInfo
3483 (14) ../DiskInfo
3484 (15) .
```

3485 If used as a "filter," this XPath expression does not filter out any instances and is the same as
3486 selecting all instances, or omitting the filter entirely. However, using the following syntax, the XPath
3487 expression selects the XML node only if the test expression in brackets evaluates to logical "true":

```
3488 (1) ../DiskInfo[LogicalDisk="C:"]
```

3489 In this case, the item is selected only if it refers to disk drive "C:"; otherwise the XML node is not
3490 selected. This XPath expression filters out all DiskInfo instances for other drives.

3491 EXAMPLE 3: Full XPath implementations may support more complex test expressions, as follows:

```
3492 (1) ../DiskInfo[CurrentMegabytes>"10" and CurrentMegabytes <"200"]
```

3493 This action selects only drives with free space within the range of values specified.

3494 In essence, the XML form of the event passes logically through the XPath processor to see if it would
3495 be selected. If so, it is delivered in the enumeration. If not, the item is discarded and not delivered as
3496 part of the enumeration.

3497 See the related clause (10.2.2) on filtering over subscriptions.

3498 **8.4 Pull**

3499 The Pull operation is initiated by sending a Pull request message to the data source. The Pull request
3500 message shall be of the following form:

```

3501 (1) <s:Envelope ...>
3502 (2)   <s:Header ...>
3503 (3)     <wsa:Action>
3504 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/Pull
3505 (5)     </wsa:Action>
3506 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
3507 (7)     <wsa:ReplyTo>wsa:EndpointReference</wsa:ReplyTo>
3508 (8)     <wsa:To>xs:anyURI</wsa:To>
3509 (9)     ...
3510 (10)  </s:Header>
3511 (11)  <s:Body ...>
3512 (12)   <wsmen:Pull ...>
3513 (13)     <wsmen:EnumerationContext>...</wsmen:EnumerationContext>
3514 (14)     <wsmen:MaxTime>xs:duration</wsmen:MaxTime> ?
3515 (15)     <wsmen:MaxElements>xs:long</wsmen:MaxElements> ?
3516 (16)     <wsmen:MaxCharacters>xs:long</wsmen:MaxCharacters> ?
3517 (17)     ...
3518 (18)   </wsmen:Pull>
3519 (19)  </s:Body>
3520 (20) </s:Envelope>

```

3521 The following describes additional, normative constraints on the preceding outline:

3522 /s:Envelope/s:Header/wsa:Action

3523 This required element shall contain the value:

3524 http://schemas.xmlsoap.org/ws/2004/09/enumeration/Pull

3525 If a SOAP Action URI is also present in the underlying transport, its value shall convey the same
3526 value.

3527 /s:Envelope/s:Body/wsmen:Pull/wsmen:EnumerationContext

3528 This required element contains the XML data that represents the current enumeration context.
3529 If the enumeration context is not valid, because it has been replaced in the response to another
3530 Pull request, it has completed (EndOfSequence has been returned in a Pull response), it has
3531 been Released, it has expired, or the data source has had to invalidate the context, then the
3532 data source should fail the request, and may generate a wsmen:InvalidEnumerationContext
3533 fault.

3534 The data source may not be able to determine that an enumeration context is not valid,
3535 especially if all of the state associated with the enumeration is kept in the enumeration context
3536 and refreshed on every PullResponse.

3537 /s:Envelope/s:Body/wsmen:Pull/wsmen:MaxTime

3538 This optional element (of type xs:duration) indicates the maximum amount of time the initiator is
3539 willing to allow the data source to assemble the Pull response. When this element is absent, the
3540 data source is not required to limit the amount of time it takes to assemble the Pull response.

3541 This is useful with data sources that accumulate elements over time and package them into a
3542 single Pull response.

3543 /s:Envelope/s:Body/wsmen:Pull/wsmen:MaxElements

3544 This optional element (of type xs:long) indicates the number of items (child elements of Items in
3545 the Pull response) the consumer is willing to accept. When this element is absent, its implied
3546 value is 1. Implementations shall not return more than this number of elements in the Pull

3547 response message. Implementations may return fewer than this number based on either the
3548 MaxTime timeout, the MaxCharacters size limit, or implementation-specific constraints.

3549 /s:Envelope/s:Body/wsmen:Pull/wsmen:MaxCharacters

3550 This optional element (of type xs:long) indicates the maximum size of the returned elements, in
3551 Unicode characters, that the initiator is willing to accept. When this element is absent, the data
3552 source is not required to limit the number of characters in the Pull response. Implementations
3553 shall not return a Pull response message whose Items element is larger than MaxCharacters.
3554 Implementations may return a smaller message based on the MaxTime timeout, the
3555 MaxElements limit, or implementation-specific constraints.

3556 Even if a Pull request contains a MaxCharacters element, the consumer shall be prepared to
3557 receive a Pull response that contains more data characters than specified, as XML
3558 canonicalization or alternate XML serialization algorithms may change the size of the
3559 representation.

3560 It may happen that the next item the data source would return to the consumer is larger than
3561 MaxCharacters. In this case, the data source may skip the item, or may return an abbreviated
3562 representation of the item that fits inside MaxCharacters. If the data source skips the item, it may
3563 return it as part of the response to a future Pull request with a larger value of MaxCharacters, or
3564 it may omit it entirely from the enumeration. If the oversize item is the last item to be returned for
3565 this enumeration context and the data source skips it, it shall include the EndOfSequence item in
3566 the Pull response and invalidate the enumeration context; that is, it may not return zero items but
3567 not consider the enumeration completed. See the discussion of EndOfSequence later in this
3568 clause.

3569 Other components of the preceding outline are not further constrained by this specification.

3570 Upon receipt of a Pull request message, the data source may wait as long as it deems necessary (but
3571 not longer than the value of the MaxTime element, if present) to produce a message for delivery to
3572 the consumer. The data source shall recognize the MaxTime element and return the
3573 wsmen:TimedOut fault if no elements are available prior to the request message's deadline.

3574 However, this fault should not cause the enumeration context to become invalid (of course, the data
3575 source may invalidate the enumeration context for other reasons). That is, the requestor should be
3576 able to issue additional Pull requests using this enumeration context after receiving this fault.

3577 Upon successful processing of a Pull request message, a data source is expected to return a Pull
3578 response message, which shall adhere to the following form:

```

3579 (1) <s:Envelope ...>
3580 (2)   <s:Header ...>
3581 (3)     <wsa:Action>
3582 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/PullResponse
3583 (5)     </wsa:Action>
3584 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
3585 (7)     <wsa:To>xs:anyURI</wsa:To>
3586 (8)     ...
3587 (9)   </s:Header>
3588 (10)  <s:Body ...>
3589 (11)   <wsmen:PullResponse ...>
3590 (12)     <wsmen:EnumerationContext>...</wsmen:EnumerationContext> ?
3591 (13)     <wsmen:Items> ?
3592 (14)       <xs:any> enumeration-specific element </xs:any> +
3593 (15)     </wsmen:Items>
3594 (16)     <wsmen:EndOfSequence/> ?
3595 (17)     ...
3596 (18)   </wsmen:PullResponse>
3597 (19) </s:Body>
3598 (20) </s:Envelope>

```

- 3599 The following describes additional, normative constraints on the preceding outline:
- 3600 /s:Envelope/s:Header/wsa:Action
- 3601 This required element shall contain the value:
- 3602 `http://schemas.xmlsoap.org/ws/2004/09/enumeration/PullResponse`
- 3603 If a SOAP Action URI is also present in the underlying transport, its value shall convey the same
- 3604 value.
- 3605 /s:Envelope/s:Body/wsmen:PullResponse/wsmen:EnumerationContext
- 3606 The optional EnumerationContext element, if present, contains a new XML representation of the
- 3607 current enumeration context. The consumer is required to replace the prior representation with
- 3608 the contents of this element.
- 3609 /s:Envelope/s:Body/wsmen:PullResponse/wsmen:Items/any
- 3610 The optional Items element contains one or more enumeration-specific elements, one for each
- 3611 element being returned.
- 3612 /s:Envelope/s:Body/wsmen:PullResponse/wsmen:EndOfSequence
- 3613 This optional element indicates that no more elements are available from this enumeration.
- 3614 Additionally, once this element is returned in a Pull response message, subsequent Pull
- 3615 requests using that enumeration context should generate an InvalidEnumerationContext fault
- 3616 message; in any case, they shall not return a valid PullResponse.
- 3617 At least one of Items or EndOfSequence shall appear. It is possible for both to appear if items are
- 3618 returned and the sequence is exhausted. Similarly, EnumerationContext and EndOfSequence shall
- 3619 not both appear; neither may appear, or one without the other, but not both in the same
- 3620 PullResponse.
- 3621 The consumer should not issue additional Pull request messages after a Pull response containing an
- 3622 EndOfSequence element has been returned. Similarly, upon receipt of a Pull response containing an
- 3623 EndOfSequence element, the consumer should not issue a Release operation to signal that the
- 3624 enumeration context is no longer needed.
- 3625 If the consumer does issue a Pull or Release on an invalid enumeration context, the result is
- 3626 undefined: the data source may ignore the request or may return an InvalidEnumerationContext fault,
- 3627 as described previously in this clause, or may take some other action.
- 3628 Because Pull allows the client to specify a wide range of batching and timing parameters, it is often
- 3629 advisable for the client to know the valid ranges ahead of time. This information can be exported from
- 3630 the service in the form of metadata, which is beyond the scope of this specification. No message-
- 3631 based negotiation is available for discovering the valid ranges of the parameters.
- 3632 Because wsman:MaxEnvelopeSize can be requested for any response in WS-Management, it is used
- 3633 in the Pull message instead of MaxCharacters, which is generally redundant and preferably is
- 3634 omitted. However, if wsman:MaxEnvelopeSize is present, it has the following characteristics:
- 3635 **R8.4-1:** If a service is exposing enumeration operations and supports Pull with the
- 3636 MaxCharacters element, the service should implement MaxCharacters as a general guideline or
- 3637 hint, but may ignore it if wsman:MaxEnvelopeSize is present, because it takes precedence. The
- 3638 service should not fault in the case of a conflict but should observe the wsman:MaxEnvelopeSize
- 3639 value.
- 3640 **R8.4-2:** If a service is exposing enumeration operations and supports Pull with the
- 3641 MaxCharacters element, and a single response element would cause the limit to be exceeded,
- 3642 the service may return the single element in violation of the hint. However, the service shall not
- 3643 violate wsman:MaxEnvelopeSize in any case.

3644 A service can send a PullResponse with fewer elements to ensure that the wsman:MaxEnvelopeSize
3645 is not exceeded. However, if a single item would cause this to be exceeded, then the rules from 6.2
3646 apply.

3647 In general, MaxCharacters is a hint, and wsman:MaxEnvelopeSize is a strict rule.

3648 **R8.4-3:** If any fault occurs during a Pull, a compliant service should allow the client to retry Pull
3649 with other parameters, such as a larger limit or with no limit, and attempt to retrieve the items.
3650 The service should not cancel the enumeration as a whole, but retain enough context to be able
3651 to retry if the client so wishes. However, the service may cancel the enumeration outright if an
3652 error occurs with an InvalidEnumerationContext fault.

3653 If a fault occurs with a Pull request, the service generally does not need to cancel the entire
3654 enumeration, but it can simply freeze the cursor and allow the client to try again.

3655 The EnumerationContext from only the latest response is considered to be valid. Although the service
3656 can return the same EnumerationContext values with each Pull, it is not required to do so and can in
3657 fact change the EnumerationContext unpredictably.

3658 **R8.4-4:** A conformant service may ignore MaxTime if wsman:OperationTimeout is also
3659 specified, as wsman:OperationTimeout takes precedence. These elements have precisely the
3660 same meaning and may be used interchangeably. If both are used, the service should observe
3661 only the wsman:OperationTimeout element.

3662 Clients can use wsman:OperationTimeout and wsman:MaxEnvelopeSize rather than MaxTime and
3663 MaxCharacters to allow for uniform message construction.

3664 Any fault issued for Pull applies to the Pull message itself, not the underlying enumeration that is in
3665 progress. The most recent EnumerationContext is still considered valid, and if the service allows a
3666 retry of the most recent Pull message, the client can continue. However, the service can terminate
3667 early upon encountering any kind of problem (as specified in R8.4-7).

3668 **R8.4-5:** This rule intentionally left blank.

3669 If no content is available, the enumerator is still considered active and the Pull message can be
3670 retried.

3671 **R8.4-6:** If a service cannot populate the PullResponse with any items before the timeout, it
3672 should return a wsman:TimedOut fault to indicate that true timeout conditions occurred and that
3673 the client is not likely to succeed by simply issuing another Pull message. If the service is only
3674 waiting for results at the point of the timeout, it should return a response with no items and an
3675 updated EnumerationContext, which may have changed, even though no items were returned, as
3676 follows:

```
3677 (1) <s:Body>  
3678 (2)   <wsmen:PullResponse>  
3679 (3)     <wsmen:EnumerationContext> ...possibly updated...  
3680 (4)     </wsmen:EnumerationContext>  
3681 (5)     <wsmen:Items/>  
3682 (6)   </wsmen:PullResponse>  
3683 (7) </s:Body>
```

3684 An empty Items block is essentially a directive from the service to try again. If the service faults with a
3685 wsman:TimedOut fault, it implies that a retry is not likely to succeed. Typically, the service knows
3686 which one to return based on its internal state. For example, on the very first Pull message, if the
3687 service is waiting for another component, a wsman:TimedOut fault could be likely. If the enumeration
3688 is continuing with no problem and after 50 requests a particular Pull message times out, the service
3689 can simply send back zero items in the expectation that the client can continue with another Pull
3690 message.

3691 **R8.4-7:** The service may terminate the entire enumeration early at any time, in which case an
 3692 InvalidEnumerationContext fault is returned. No further operations are possible, including
 3693 Release. In specific cases, such as internal errors or responses that are too large, other faults
 3694 may also be returned. In all such cases, the service should invalidate the enumeration context as
 3695 well.

3696 **R8.4-8:** If the EndOfSequence marker occurs in the PullResponse message, the
 3697 EnumerationContext element shall be omitted, as the enumeration has completed. The client
 3698 cannot subsequently issue a Release message.

3699 Normally, the end of an enumeration in all cases is reported by the EndOfSequence element being
 3700 present in the PullResponse content, not through faults. If the client attempts to enumerate past the
 3701 end of an enumeration, an InvalidEnumerationContext fault is returned. The client need not issue a
 3702 Release message if the EndOfSequence actually occurs because the enumeration is then completed
 3703 and the enumeration context is invalid.

3704 **R8.4-9:** If no MaxElements element is specified, the batch size is 1.

3705 **R8.4-10:** If the value of MaxElements is larger than the service supports, the service may ignore
 3706 the value and use any default maximum of its own.

3707 The service can export its maximum MaxElements value in metadata, but the format and location of
 3708 such metadata is beyond the scope of this specification.

3709 **R8.4-11:** The EnumerationContext element shall be present in all Pull requests, even if the
 3710 service uses a constant value for the lifetime of the enumeration sequence.

3711 8.5 Release

3712 The Release operation is initiated by sending a Release request message to the data source. The
 3713 Release request message shall be of the following form:

```

3714 (1) <s:Envelope ...>
3715 (2)   <s:Header ...>
3716 (3)     <wsa:Action>
3717 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/Release
3718 (5)     </wsa:Action>
3719 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
3720 (7)     <wsa:ReplyTo>wsa:EndpointReference</wsa:ReplyTo>
3721 (8)     <wsa:To>xs:anyURI</wsa:To>
3722 (9)     ...
3723 (10)  </s:Header>
3724 (11)  <s:Body ...>
3725 (12)    <wsmen:Release ...>
3726 (13)      <wsmen:EnumerationContext>...</wsmen:EnumerationContext>
3727 (14)    ...
3728 (15)  </wsmen:Release>
3729 (16)  </s:Body>
3730 (17) </s:Envelope>
  
```

3731 The following describes additional, normative constraints on the preceding outline:

3732 /s:Envelope/s:Header/wsa:Action

3733 This required element shall contain the value:

3734 <http://schemas.xmlsoap.org/ws/2004/09/enumeration/Release>

3735 If a SOAP Action URI is also present in the underlying transport, its value shall convey the same
3736 value.

3737 /s:Envelope/s:Body/wsmen:Release/wsmen:EnumerationContext

3738 This required element contains the XML data that represents the enumeration context being
3739 abandoned.

3740 Other components of the preceding outline are not further constrained by this specification.

3741 Upon successful processing of a Release request message, a data source is expected to return a
3742 Release response message, which shall adhere to the following form:

```
3743 (1) <s:Envelope ...>
3744 (2)   <s:Header ...>
3745 (3)     <wsa:Action>
3746 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/ReleaseResponse
3747 (5)     </wsa:Action>
3748 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
3749 (7)     <wsa:To>xs:anyURI</wsa:To>
3750 (8)     ...
3751 (9)   </s:Header>
3752 (10)  <s:Body />
3753 (11) </s:Envelope>
```

3754 The following describes additional, normative constraints on the preceding outline:

3755 /s:Envelope/s:Header/wsa:Action

3756 This required element shall contain the value:

3757 <http://schemas.xmlsoap.org/ws/2004/09/enumeration/ReleaseResponse>

3758 If a SOAP Action URI is also present in the underlying transport, its value shall convey the same
3759 value.

3760 Release is used only to perform an early cancellation of the enumeration. In cases in which it is not
3761 actually needed, the implementation can expose a dummy implementation that always succeeds.
3762 This promotes uniform client-side messaging.

3763 **R8.5-1:** The service shall recognize and process the Release message if the enumeration is
3764 terminated early. If an EndOfSequence marker occurs in a PullResponse message, the
3765 enumerator is already completed and a Release message cannot be issued because no up-to-
3766 date EnumerationContext exists.

3767 **R8.5-2:** The client may fail to deliver the Release message in a timely fashion or may never
3768 send it. A conformant service may terminate the enumeration after a suitable idle time has
3769 expired, and any attempt to reuse the enumeration context shall result in an
3770 InvalidEnumerationContext fault.

3771 **R8.5-3:** This rule intentionally left blank.

3772 **R8.5-4:** The service may accept a Release message asynchronously to any Pull requests
3773 already in progress and cancel the enumeration. The service may refuse such an asynchronous
3774 request and fault it with a wsmen:UnsupportedFeature fault with the following detail code:

3775 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AsynchronousRequest>

3776 The service may also queue or block the request and serialize it so that it is processed after the Pull
3777 message.

3778 In most cases, it is desirable to be able to asynchronously cancel an outstanding Pull message. This
 3779 capability requires the service to be able to receive the Release message asynchronously while still
 3780 processing a pending Pull message. Further, it requires that the EnumerationContext element contain
 3781 information that is constant between Pull operations.

3782 NOTE: If the value of EnumerationContext is a simple increasing integer, Release always uses a previous value,
 3783 so the service may consider it to be invalid. If the EnumerationContext element contains a value that is constant
 3784 across Pull requests (as well as any other information that the service might need), the service can more easily
 3785 implement the cancellation.

3786 8.6 Ad-Hoc Queries and Fragment-Level Enumerations

3787 As discussed in 7.7, it is desirable that clients be able to access subsets of a representation. This is
 3788 especially important in the area of query processing, where users routinely want to execute XPath or
 3789 XQuery operations over the representation to receive ad-hoc results.

3790 Because SOAP messages need to conform to known schemas, and ad-hoc queries return results
 3791 that are dynamically generated and might conform to no schema, the wsman:XmlFragment wrapper
 3792 from 7.7 is used to wrap the responses.

3793 **R8.6-1:** The service may support ad-hoc compositional queries, projections, or enumerations of
 3794 fragments of the representation objects by supplying a suitable dialect in the wsman:Filter. The
 3795 resulting set of Items in the PullResponse element (or EnumerateResponse element if
 3796 OptimizedEnumeration is used) should be wrapped with wsman:XmlFragment wrappers as
 3797 follows:

```
3798 (1) <s:Body>
3799 (2)   <wsmen:PullResponse>
3800 (3)     <wsmen:EnumerationContext> ..possibly updated..
3801 (4)   </wsmen:EnumerationContext>
3802 (5)   <wsmen:Items>
3803 (6)     <wsman:XmlFragment>
3804 (7)       XML content
3805 (8)     </wsman:XmlFragment>
3806 (9)     <wsman:XmlFragment>
3807 (10)      XML content
3808 (11)    </wsman:XmlFragment>
3809 (12)    ...
3810 (13)   </wsmen:Items>
3811 (14)  </wsmen:PullResponse>
3812 (15) </s:Body>
```

3813 The schema for wsman:XmlFragment contains a directive to suppress schema validation, allowing a
 3814 validating parser to accept ad-hoc content produced by the query processor acting behind the
 3815 enumeration.

3816 [XPath 1.0](#) and [XQuery 1.0](#) already support returning subsets or compositions of representations, so
 3817 they are suitable for use in this regard.

3818 **R8.6-2:** If the service does not support fragment-level enumeration, it should return a
 3819 wsmen:FilterDialectRequestedUnavailable fault, the same as for any other unsupported dialect.

3820 The XPath expression used for filtering is still as described in the Enumeration clauses (see 8.2,
 3821 8.2.2, 8.2.3). The wsman:XmlFragment wrappers are applied after the XPath is evaluated to prevent
 3822 schema violations if the XPath selects node sets that are fragments and not legal according to the
 3823 original schema.

3824 **8.7 Enumeration of EPRs**

3825 Typically, inferring the EPR of an enumerated object simply by inspection is not possible. In many
 3826 cases, it is desirable to enumerate the EPRs of objects rather than the objects themselves. Such
 3827 EPRs can be usable in subsequent Get or Delete requests, for example. Similarly, it is often desirable
 3828 to enumerate both the objects and the associated EPRs.

3829 The default behavior for Enumerate is as defined in 8.1. However, WS-Management provides an
 3830 additional extension for controlling the output of the enumeration.

3831 **R8.7-1:** A service may optionally support the wsman:EnumerationMode modifier element with a
 3832 value of *EnumerateEPR*, which returns only the EPRs of the objects as the result of the
 3833 enumeration.

3834 **EXAMPLE 1:**

```
3835 (1) <s:Envelope ...>
3836 (2)   <s:Header>
3837 (3)     ...
3838 (4)     <wsa:Action>
3839 (5)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate
3840 (6)     </wsa:Action>
3841 (7)     ...
3842 (8)   </s:Header>
3843 (9)   <s:Body>
3844 (10)    <wsmen:Enumerate>
3845 (11)      <wsman:Filter Dialect="..."> filter </wsman:Filter>
3846 (12)      <wsman:EnumerationMode> EnumerateEPR </wsman:EnumerationMode>
3847 (13)      ...
3848 (14)    </wsmen:Enumerate>
3849 (15)  </s:Body>
3850 (16) </s:Envelope>
```

3851 **EXAMPLE 2:** The hypothetical response would appear as in the following example:

```
3852 (17) <s:Body>
3853 (18)   <wsmen:PullResponse>
3854 (19)     <wsmen:Items>
3855 (20)       <wsa:EndpointReference> ... </wsa:EndpointReference>
3856 (21)       <wsa:EndpointReference> ... </wsa:EndpointReference>
3857 (22)       <wsa:EndpointReference> ... </wsa:EndpointReference>
3858 (23)       ...
3859 (24)     </wsmen:Items>
3860 (25)   </wsmen:PullResponse>
3861 (26) </s:Body>
```

3862 The filter, if any, is still applied to the enumeration, but the response contains only the EPRs of the
 3863 items that would have been returned. These EPRs are intended for use in subsequent Get
 3864 operations.

3865 **R8.7-2:** A service may optionally support the wsman:EnumerationMode modifier with the value
 3866 of *EnumerateObjectAndEPR*. If present, the enumerated objects are wrapped in a wsman:Item
 3867 element that juxtaposes two XML representations: the payload representation followed by the
 3868 associated wsa:EndpointReference.

3869 **EXAMPLE 3:** The wsman:EnumerationMode example appears as follows:

```
3870 (1) <s:Header>
3871 (2)   ...
3872 (3)   <wsa:Action>
```

```

3873 (4) http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate
3874 (5) </wsa:Action>
3875 (6) </s:Header>
3876 (7) <s:Body>
3877 (8) <wsmen:Enumerate>
3878 (9) <wsman:Filter Dialect="..."> filter </wsman:Filter>
3879 (10) <wsman:EnumerationMode> EnumerateObjectAndEPR
3880 </wsman:EnumerationMode>
3881 (11) ...
3882 (12) </wsmen:Enumerate>
3883 (13) </s:Body>

```

3884 **EXAMPLE 4:** The response appears as follows:

```

3885 (1) <s:Body>
3886 (2) <wsmen:PullResponse>
3887 (3) <wsmen:Items>
3888 (4) <wsman:Item>
3889 (5) <PayloadObject xmlns="..."> ... </PayloadObject> <!-- Object -->
3890 (6) <wsa:EndpointReference> ... </wsa:EndpointReference> <!-- EPR -->
3891 (7) </wsman:Item>
3892 (8) <wsman:Item>
3893 (9) <PayloadObject xmlns="..."> ... </PayloadObject> <!-- Object -->
3894 (10) <wsa:EndpointReference> ... </wsa:EndpointReference> <!-- EPR -->
3895 (11) </wsman:Item>
3896 (12) ...
3897 (13) </wsmen:Items>
3898 (14) </wsmen:PullResponse>
3899 (15) </s:Body>

```

3900 In the preceding example, each item is wrapped in a wsman:Item wrapper (line 8), which itself contains the
3901 representation object (line 9) followed by its EPR (line 10). As many wsman:Item objects may be present as is
3902 consistent with other encoding limitations.

3903 **R8.7-3:** If a service does not support the wsman:EnumerationMode modifier, it shall return a
3904 fault of wsman:UnsupportedFeature with the following detail code:

3905 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EnumerationMode>

3906 8.8 Renew

3907 To renew an enumeration, the consumer sends a request of the following form to the data source:

```

3908 (1) <s:Envelope ...>
3909 (2) <s:Header ...>
3910 (3) <wsa:Action>
3911 (4) http://schemas.xmlsoap.org/ws/2004/09/enumeration/Renew
3912 (5) </wsa:Action>
3913 (6) <wsa:MessageID>xs:anyURI</wsa:MessageID>
3914 (7) <wsa:FaultTo>endpoint-reference</wsa:FaultTo> ?
3915 (8) <wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
3916 (9) <wsa:To>xs:anyURI</wsa:To>
3917 (10) ...
3918 (11) </s:Header>
3919 (12) <s:Body ...>
3920 (13) <wsmen:Renew ...>
3921 (14) <wsmen:EnumerationContext>...</wsmen:EnumerationContext>
3922 (15) <wsmen:Expires>[xs:dateTime | xs:duration]</wsmen:Expires> ?
3923 (16) ...
3924 (17) </wsmen:Renew>

```

3925 (18) </s:Body>
 3926 (19) </s:Envelope>

3927 Components of the preceding outline are additionally constrained as for a request to create an
 3928 enumeration with the following addition(s):

3929 /s:Envelope/s:Body/*/wsmen:EnumerationContext

3930 This required element contains the XML data that represents the current enumeration context.

3931 If the enumeration context is not valid, either because it has been replaced in the response to
 3932 another Pull request, or because it has completed (EndOfSequence has been returned in a Pull
 3933 response), or because it has been Released, or because it has expired, or because the data
 3934 source has had to invalidate the context, then the data source should fail the request, and may
 3935 generate a wsmen:InvalidEnumerationContext fault.

3936 The data source may not be able to determine that an enumeration context is not valid,
 3937 especially if all of the state associated with the enumeration is kept in the enumeration context
 3938 and refreshed on every PullResponse.

3939 Other components of the preceding outline are not further constrained by this specification.

3940 If the data source accepts a request to renew an enumeration, it shall reply with a response of the
 3941 following form:

```

3942 (1) <s:Envelope ...>
3943 (2)   <s:Header ...>
3944 (3)     <wsa:Action>
3945 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/RenewResponse
3946 (5)     </wsa:Action>
3947 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
3948 (7)     <wsa:To>xs:anyURI</wsa:To>
3949 (8)     ...
3950 (9)   </s:Header>
3951 (10)  <s:Body ...>
3952 (11)   <wsmen:RenewResponse ...>
3953 (12)     <wsmen:Expires>[xs:dateTime | xs:duration]</wsmen:Expires> ?
3954 (13)     <wsmen:EnumerationContext>...</wsmen:EnumerationContext> ?
3955 (14)     ...
3956 (15)   </wsmen:RenewResponse>
3957 (16)  </s:Body>
3958 (17) </s:Envelope>
```

3959 Components of the preceding outline listed are constrained as for a response to an Enumerate
 3960 request with the following addition:

3961 /s:Envelope/s:Body/wsmen:RenewResponse/wsmen:Expires

3962 If the requested expiration is a duration, then the implied start of that duration is the time when
 3963 the data source starts processing the Renew request.

3964 /s:Envelope/s:Body/wsmen:RenewResponse/wsmen:EnumerationContext

3965 This element is optional in this response.

3966 If the data source chooses not to renew this enumeration, the request shall fail, and the data
 3967 source should generate a wsmen:UnableToRenew fault indicating that the renewal was not
 3968 accepted.

3969 Other components of the preceding outline are not further constrained by this specification.

3970 **8.9 GetStatus**

3971 To get the status of an enumeration, the subscriber sends a request of the following form to the data
3972 source:

```

3973 (1) <s:Envelope ...>
3974 (2)   <s:Header ...>
3975 (3)     <wsa:Action>
3976 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatus
3977 (5)     </wsa:Action>
3978 (6)     <wsa:MessageID>xs:anyURI</wsa:MessageID>
3979 (7)     <wsa:FaultTo>endpoint-reference</wsa:FaultTo> ?
3980 (8)     <wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
3981 (9)     <wsa:To>xs:anyURI</wsa:To>
3982 (10)    ...
3983 (11)   </s:Header>
3984 (12)   <s:Body ...>
3985 (13)     <wsmen:GetStatus ...>
3986 (14)       <wsmen:EnumerationContext>...</wsmen:EnumerationContext> ?
3987 (15)     ...
3988 (16)   </wsmen:GetStatus>
3989 (17)   </s:Body>
3990 (18) </s:Envelope>

```

3991 Components of the preceding outline are additionally constrained as for a request to renew an
3992 enumeration. Other components of the preceding outline are not further constrained by this
3993 specification.

3994 If the enumeration is valid and has not expired, the data source shall reply with a response of the
3995 following form:

```

3996 (1) <s:Envelope ...>
3997 (2)   <s:Header ...>
3998 (3)     <wsa:Action>
3999 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatusResponse
4000 (5)     </wsa:Action>
4001 (6)     <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
4002 (7)     <wsa:To>xs:anyURI</wsa:To>
4003 (8)     ...
4004 (9)   </s:Header>
4005 (10)  <s:Body ...>
4006 (11)   <wsmen:GetStatusResponse ...>
4007 (12)     <wsmen:Expires>[xs:dateTime | xs:duration]</wsmen:Expires> ?
4008 (13)   ...
4009 (14)   </wsmen:GetStatusResponse>
4010 (15)   </s:Body>
4011 (16) </s:Envelope>

```

4012 Components of the preceding outline are constrained as for a response to a Renew request. Other
4013 components of the preceding outline are not further constrained by this specification.

4014 **8.10 EnumerationEnd**

4015 If the data source terminates an enumeration unexpectedly, the data source should send an
4016 EnumerationEnd SOAP message to the endpoint reference indicated when the enumeration was
4017 created. The message shall be of the following form:

```

4018 (1) <s:Envelope ...>
4019 (2)   <s:Header ...>
4020 (3)     <wsa:Action>
4021 (4)       http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerationEnd

```

```

4022     (5)     </wsa:Action>
4023     (6)     <wsa:To>xs:anyURI</wsa:To>
4024     (7)     ...
4025     (8)     </s:Header>
4026     (9)     <s:Body ...>
4027     (10)    <wsmen:EnumerationEnd ...>
4028     (11)    <wsmen:EnumerationContext>...</wsmen:EnumerationContext>
4029     (12)    <wsmen:Code>
4030     (13)    [
4031     (14)    http://schemas.xmlsoap.org/ws/2004/09/enumeration/SourceShuttingDown
4032     (15)    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/SourceCancelling
4033     (16)    ]
4034     (17)    </wsmen:Code>
4035     (18)    <wsmen:Reason xml:lang="language identifier" >
4036     (19)    xs:string
4037     (20)    </wsmen:Reason> ?
4038     (21)    ...
4039     (22)    </wsmen:EnumerationEnd>
4040     (23)    </s:Body>
4041     (24)    </s:Envelope>

```

4042 The following describes additional, normative constraints on the preceding outline:

4043 /s:Envelope/s:Body/wsmen:Release/wsmen:EnumerationContext

4044 This required element contains the XML data that represents the enumeration context being
4045 terminated. It is recommended that consumers DO NOT attempt to compare this element
4046 against any collection of wsmen:EnumerationContext elements for purposes of correlation,
4047 because that requires the ability to compare arbitrary XML elements. If consumers wish to
4048 correlate this message against their outstanding contexts, it is recommend that they use the
4049 reference parameters of the /wsmen:Enumerate/wsmen:EndTo EPR.

4050 /s:Envelope/s:Body/wsmen:EnumerationEnd/wsmen:Code =
4051 "http://schemas.xmlsoap.org/ws/2004/09/enumeration/SourceShuttingDown"

4052 This value shall be used if the data source terminated the enumeration because the source is
4053 being shut down in a controlled manner; that is, if the data source is being shut down but has the
4054 opportunity to send an EnumerationEnd message before it exits.

4055 /s:Envelope/s:Body/wsmen:EnumerationEnd/wsmen:Code =
4056 "http://schemas.xmlsoap.org/ws/2004/09/enumeration/SourceCancelling"

4057 This value shall be used if the data source terminated the enumeration for some other reason
4058 before it expired.

4059 /s:Envelope/s:Body/wsmen:EnumerationEnd/wsmen:Reason

4060 This optional element contains text, in the language specified by the @xml:lang attribute,
4061 describing the reason for the unexpected enumeration termination.

4062 Other components of the preceding outline are not further constrained by this specification.

4063 9 Custom Actions (Methods)

4064 Custom actions, or "methods," are ordinary SOAP messages with unique Actions. An implementation
4065 can support resource-specific methods in any form, subject to the addressing model and restrictions
4066 described in clause 5 of this specification.

4067 **R9-1:** A conformant service may expose any custom actions or methods.

4068 **R9-2:** If custom methods are exported, Addressing rules, as described elsewhere in this
4069 specification, shall be observed, and each custom method shall have a unique wsa:Action.

4070 **R9-3:** If a request does not contain the correct parameters for the custom action, the service
4071 may return a wsman:InvalidParameter fault. Fault details for incorrect type and incorrect name
4072 may also be included.

4073 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/TypeMismatch` (incorrect type)

4074 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidName` (incorrect name)

4075 As defined by Addressing, the Action URI is used to describe the semantics of the operation and the
4076 wsa:To element describes the destination of the message. A custom method thus has a dedicated
4077 Addressing Action URI.

4078 Because options are a parameterization technique for message types that are not user-extensible,
4079 such as the resource access operations, they are not appropriate for use as a custom method or
4080 combined with a custom method. Custom operations defined in a WSDL document define any
4081 required parameters and thus expose naming and type checking in a stringent way. Mixing
4082 wsman:OptionSet with a strongly typed WSDL operation is likely to lead to confusion.

4083 **10 Notifications (Eventing)**

4084 **10.1 General**

4085 Management infrastructures often want to receive messages when events occur in remote
4086 management services and applications. A mechanism for registering interest is needed because the
4087 set of Web services interested in receiving such messages is often unknown in advance or changes
4088 over time. This specification defines a set of operations for one management Web service (called a
4089 "subscriber") to register interest (called a "subscription") with another management Web service
4090 (called an "event source") in receiving messages about events (called "notifications" or "event
4091 messages"). The subscriber may manage the subscription by interacting with a Web service (called
4092 the "subscription manager") designated by the event source.

4093 To improve robustness, a subscription may be leased by an event source to a subscriber, and the
4094 subscription expires over time. The subscription manager provides the ability for the subscriber to
4095 renew or cancel the subscription before it expires.

4096 There are many mechanisms by which event sources may deliver events to event sinks. This
4097 specification provides an extensible way for subscribers to identify the delivery mechanism they
4098 prefer. While asynchronous, pushed delivery is defined here; the intent is that there should be no
4099 limitation or restriction on the delivery mechanisms capable of being supported by this specification.

4100 To create, renew, and delete subscriptions, subscribers send request messages to event sources and
4101 subscription managers.

4102 When an event source accepts a request to create a subscription, it typically does so for a given
4103 amount of time, although an event source may accept an indefinite subscription with no time-based
4104 expiration. If the subscription manager accepts a renewal request, it updates that amount of time.
4105 During that time, notifications are delivered by the event source to the requested event sink. An event
4106 source may support filtering to limit notifications that are delivered to the event sink; if it does, and a
4107 subscribe request contains a filter, the event source sends only notifications that match the requested
4108 filter. The event source sends notifications until one of the following happens: the subscription
4109 manager accepts an unsubscribe request for the subscription, the subscription expires without being
4110 renewed, or the event source cancels the subscription prematurely. In this last case, the event source
4111 makes a best effort to indicate why the subscription ended.

4112 In the absence of reliable messaging at the application layer (for example, [WS-ReliableMessaging]),
 4113 messages defined herein are delivered using the quality of service of the underlying transport(s) and
 4114 on a best-effort basis at the application layer.

4115 If a managed entity emits events, it can publish those events using this publish-and-subscribe
 4116 mechanism and paradigms.

4117 **R10.1-1:** If a resource can emit events and allows clients to subscribe to and receive notification
 4118 messages, it shall do so by implementing the operations as specified in this clause.

4119 **R10.1-2:** If the eventing mechanism as described in this clause is supported, the
 4120 wsme:Subscribe, wsme:Renew, and wsme:Unsubscribe messages shall be supported. The
 4121 wsme:SubscriptionEnd message is optional. The wsme:GetStatus message in a constrained
 4122 environment is a candidate for exclusion. If this message is not supported, then a
 4123 wsa:ActionNotSupported fault shall be returned in response to this request.

4124 10.2 Subscribe

4125 In some scenarios the event source itself manages the subscriptions it has created. In other
 4126 scenarios, for example a geographically distributed publish-and-subscribe system, it may be useful to
 4127 delegate the management of a subscription to another Web service. To support this flexibility, the
 4128 response to a subscription request to an event source includes the EPR of a service that the
 4129 subscriber may interact with to manage this subscription. This EPR should be the target for future
 4130 requests to renew or cancel the subscription. It may address the same Web service (Address and
 4131 ReferenceParameters) as the event source itself, or it may address some other Web service to which
 4132 the event source has delegated management of this subscription; however, the full subscription
 4133 manager EPR (Address and ReferenceParameters) must be unique for each subscription.

4134 We use the term "subscription manager" in this specification to refer to the Web service that manages
 4135 the subscription, whether it is the event source itself or some separate Web service.

4136 To create a subscription, a subscriber sends a request message of the following form to an event
 4137 source:

```

4138 (1) <s:Envelope ...>
4139 (2)   <s:Header ...>
4140 (3)     <wsa:Action>
4141 (4)       http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
4142 (5)     </wsa:Action>
4143 (6)     ...
4144 (7)   </s:Header>
4145 (8)   <s:Body ...>
4146 (9)     <wsme:Subscribe ...>
4147 (10)      <wsme:EndTo>endpoint-reference</wsme:EndTo> ?
4148 (11)      <wsme:Delivery Mode="xs:anyURI"? >xs:any</wsme:Delivery>
4149 (12)      <wsme:Expires>[xs:dateTime | xs:duration]</wsme:Expires> ?
4150 (13)      <wsme:Filter Dialect="xs:anyURI"? > xs:any </wsme:Filter> ?
4151 (14)      ...
4152 (15)    </wsme:Subscribe>
4153 (16)  </s:Body>
4154 (17) </s:Envelope>
  
```

4155 The following describes additional, normative constraints on the preceding outline:

4156 /s:Envelope/s:Header/wsa:Action

4157 If a SOAP Action URI is used in the binding for SOAP, the value indicated herein shall be used
 4158 for that URI.

- 4159 /s:Envelope/s:Body*/wsme:EndTo
- 4160 Where to send a SubscriptionEnd message if the subscription is terminated unexpectedly. If
4161 present, this element shall be of type wsa:EndpointReferenceType. The default is not to send
4162 this message. The endpoint referenced by this EPR shall implement a binding of the
4163 "EndToEndPoint" portType described in ANNEX I.
- 4164 /s:Envelope/s:Body*/wsme:Delivery
- 4165 A delivery destination for notification messages, using some delivery mode.
- 4166 /s:Envelope/s:Body*/wsme:Delivery/@Mode
- 4167 The delivery mode to be used for notification messages sent in relation to this subscription.
4168 Implied value is "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push", which
4169 indicates that Push Mode delivery should be used.
- 4170 If the event source does not support the requested delivery mode, the request shall fail, and the
4171 event source may generate a wsme:DeliveryModeRequestedUnavailable fault indicating that the
4172 requested delivery mode is not supported.
- 4173 /s:Envelope/s:Body*/wsme:Delivery/@Mode="http://schemas.xmlsoap.org/ws/2004/08/eventing/Deliv
4174 eryModes/Push"
- 4175 The value of /s:Envelope/s:Body*/wsme:Delivery is a single element, NotifyTo, that contains the
4176 endpoint reference to which notification messages should be sent.
- 4177 /s:Envelope/s:Body*/wsme:Expires
- 4178 Requested expiration time for the subscription. (No implied value.) The event source defines the
4179 actual expiration and is not constrained to use a time less or greater than the requested
4180 expiration. The expiration time may be a specific time or a duration from the subscription's
4181 creation time. Both specific times and durations are interpreted based on the event source's
4182 clock.
- 4183 If this element does not appear, then the request is for a subscription that will not expire. That is,
4184 the subscriber is requesting the event source to create a subscription with an indefinite lifetime. If
4185 the event source grants such a subscription, it may be terminated by the subscriber using an
4186 Unsubscribe request, or it may be terminated by the event source at any time for reasons such
4187 as connection termination, resource constraints, or system shut-down.
- 4188 If the expiration time is either a zero duration or a specific time that occurs in the past according
4189 to the event source, then the request shall fail, and the event source may generate a
4190 InvalidExpirationTime fault indicating that an invalid expiration time was requested.
- 4191 Some event sources may not have a "wall time" clock available, and so are only able to accept
4192 durations as expirations. If such a source receives a Subscribe request containing a specific time
4193 expiration, then the request may fail; if so, the event source may generate an
4194 UnsupportedExpirationType fault indicating that an unsupported expiration type was requested.
- 4195 /s:Envelope/s:Body*/wsme:Filter
- 4196 A Boolean expression in some dialect, either as a string or as an XML fragment. If the
4197 expression evaluates to false for a notification, the notification shall not be sent to the event sink.
4198 Implied value is an expression that always returns true. If the event source does not support
4199 filtering, then a request that specifies a filter shall fail, and the event source may generate a
4200 wsme:FilteringNotSupported fault indicating that filtering is not supported.

- 4201 If the event source supports filtering but cannot honor the requested filtering, the request shall
 4202 fail, and the event source may generate a wsme:FilteringRequestedUnavailable fault indicating
 4203 that the requested filter dialect is not supported.
- 4204 /s:Envelope/s:Body*/wsme:Filter/@Dialect
- 4205 Implied value is "http://www.w3.org/TR/1999/REC-xpath-19991116".
- 4206 While an XPath predicate expression provides great flexibility and power, alternate filter dialects
 4207 may be defined. For instance, a simpler, less powerful dialect might be defined for resource-
 4208 constrained implementations, or a new dialect might be defined to support filtering based on data
 4209 not included in the notification message itself. If desired, a filter dialect could allow the definition
 4210 of a composite filter that contained multiple filters from other dialects.
- 4211 /s:Envelope/s:Body*/wsme:Filter/@Dialect=" http://www.w3.org/TR/1999/REC-xpath-19991116"
- 4212 Value of /s:Envelope/s:Body*/wsme:Filter is an XPath [XPath 1.0](#) predicate expression
 4213 (PredicateExpr); the context of the expression is:
- 4214 • **Context Node:** the SOAP Envelope containing the notification
 - 4215 • **Context Position:** 1
 - 4216 • **Context Size:** 1
 - 4217 • **Variable Bindings:** None
 - 4218 • **Function Libraries:** Core Function Library [XPath 1.0](#)
 - 4219 • **Namespace Declarations:** The [in-scope namespaces] property [XML Infoset](#) of
 4220 /s:Envelope/s:Body*/wsme:Filter
- 4221 Other message information headers defined by Addressing may be included in the request and
 4222 response messages, according to the usage and semantics defined in Addressing.
- 4223 Other components of the preceding outline are not further constrained by this specification.
- 4224 If the event source accepts a request to create a subscription, it shall reply with a response of the
 4225 following form:
- ```

4226 (1) <s:Envelope ...>
4227 (2) <s:Header ...>
4228 (3) <wsa:Action>
4229 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscribeResponse
4230 (5) </wsa:Action>
4231 (6) ...
4232 (7) </s:Header>
4233 (8) <s:Body ...>
4234 (9) <wsme:SubscribeResponse ...>
4235 (10) <wsme:SubscriptionManager>
4236 (11) wsa:EndpointReferenceType
4237 (12) </wsme:SubscriptionManager>
4238 (13) <wsme:Expires>[xs:dateTime | xs:duration]</wsme:Expires>
4239 (14) ...
4240 (15) </wsme:SubscribeResponse>
4241 (16) </s:Body>
4242 (17) </s:Envelope>

```

- 4243 The following describes additional, normative constraints on the preceding outline:
- 4244 `/s:Envelope/S:Header/wsa:RelatesTo`
- 4245 Shall be the value of the `wsa:MessageID` of the corresponding request.
- 4246 `/s:Envelope/s:Body/*/wsme:SubscriptionManager`
- 4247 The EPR of the subscription manager for this subscription.
- 4248 In some cases, it is convenient for all EPRs issued by a single event source to address a single  
4249 Web service and use a reference parameter to distinguish among the active subscriptions. For  
4250 convenience in this common situation, this specification defines a global element, Identifier of  
4251 type `xs:anyURI`, that may be used as a distinguishing reference parameter if desired by the  
4252 event source.
- 4253 `/s:Envelope/s:Body/*/wsme:Expires`
- 4254 The expiration time assigned by the event source. The expiration time may be either an absolute  
4255 time or a duration but should be of the same type as the requested expiration (if any).
- 4256 If this element does not appear, then the subscription will not expire. That is, the subscription  
4257 has an indefinite lifetime. It may be terminated by the subscriber using an Unsubscribe request,  
4258 or it may be terminated by the event source at any time for reasons such as connection  
4259 termination, resource constraints, or system shut-down.
- 4260 Other components of the preceding outline are not further constrained by this specification.
- 4261 If the event source chooses not to accept a subscription, the request shall fail, and the event source  
4262 may generate a `wsme:EventSourceUnableToProcess` fault indicating that the request was not  
4263 accepted.
- 4264 This specification does not constrain notifications because any message may be a notification.
- 4265 However, if a subscribing event sink wishes to have notifications specifically marked, it may specify  
4266 literal SOAP header blocks in the Subscribe request, in the  
4267 `/s:Envelope/s:Body/wsme:Subscribe/wsme:NotifyTo/wsa:ReferenceParameters` elements; per  
4268 Addressing, the event source shall include each such literal SOAP header block in every notification  
4269 sent to the endpoint addressed by `/s:Envelope/s:Body/wsme:Subscribe/wsme:NotifyTo`.
- 4270 **10.2.1 General**
- 4271 WS-Management uses Subscribe substantially as documented here, except that the  
4272 WS-Management default addressing model is incorporated as described in 5.1.
- 4273 **R10.2.1-1:** The identity of the event source shall be based on the Addressing EPR.
- 4274 **R10.2.1-2:** If the service cannot support the requested addressing, it should return a  
4275 `wsman:UnsupportedFeature` fault with the following detail code:
- 4276 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode`
- 4277 Verifying that the address is usable allows errors to be detected at the time the subscription is  
4278 created. For example, if the address cannot be reached due to firewall configuration and the service  
4279 can detect this, telling the client allows for it to be corrected immediately.
- 4280 **R10.2.1-3:** Because many delivery modes require a separate connection to deliver the event,  
4281 the service should comply with the security profiles defined in clause 11 of this specification, if  
4282 HTTP or HTTPS is used to deliver events. If no security is specified, the service may attempt to

4283 use default security mechanisms, or return a wsman:UnsupportedFeature fault with the following  
4284 detail code:

4285 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsecureAddress`

4286 Because clients might need to have client-side context sent back with each event delivery, the  
4287 NotifyTo address in the Delivery block can be used for this purpose. This NotifyTo EPR can contain  
4288 any number of client-defined reference parameters.

4289 **R10.2.1-4:** A service may validate the address by attempting a connection while the Subscribe  
4290 request is being processed to ensure delivery can occur successfully. If the service determines  
4291 that the address is not valid or permissions cannot be acquired, it should emit a  
4292 wsman:EventDeliverToUnusable fault.

4293 This situation can occur when the address is incorrect or when the event source cannot acquire  
4294 permissions to deliver events properly.

4295 **R10.2.1-5:** Any reference parameters supplied in the NotifyTo address shall be included with  
4296 each event delivery as top-level headers as specified 5.4. If EndTo is supported, this behavior  
4297 applies as well.

4298 When the default addressing model is used by the service, the ResourceURI is often used to  
4299 reference the logical event source, and selector values can additionally be used to indicate a real or  
4300 virtual log within the scope of that source, or might even be used to limit the types or groups of events  
4301 available. This action can logically overlap with the Filter mechanism in the subscription body itself, so  
4302 due consideration should be given to the interplay among the address of the event source, the types  
4303 of events it can publish, and the subscription-level filtering.

4304 If a client needs to have events delivered to more than one destination, more than one subscription is  
4305 required.

4306 **R10.2.1-6:** If the events contain localized content, the service should accept a subscription with  
4307 a wsman:Locale block acting as a hint (see 6.3) within the Delivery block of the Subscribe  
4308 message. The language is encoded in an xml:lang attribute using [RFC 5646](#) language codes.

4309 The service attempts to localize any descriptive content to the specified language when delivering  
4310 such events, which is outlined as follows:

```
4311 (1) <wsme:Subscribe>
4312 (2) <wsme:Delivery>
4313 (3) <wsme:NotifyTo> ... </wsme:NotifyTo>
4314 (4) <wsman:Locale xml:lang="language-code"/>
4315 (5) </wsme:Delivery>
4316 (6) </wsme:Subscribe>
```

4317 NOTE: In this context, the wsman:Locale element (defined in 6.3) is not a SOAP header and mustUnderstand  
4318 cannot be used.

4319 **R10.2.1-7:** The service should accept a subscription with a wsman:ContentEncoding block  
4320 within the Delivery block of the Subscribe message. This block acts as a hint to indicate how the  
4321 delivered events are to be encoded. The two standard xs:language tokens defined for this  
4322 purpose are "UTF-8" or "UTF-16", although other encoding formats may be specified if  
4323 necessary. The service should attempt to encode the events using the requested language token,  
4324 as in the following example:

4325 EXAMPLE:

```
4326 (1) <wsme:Subscribe>
4327 (2) <wsme:Delivery>
4328 (3) ...
```

```

4329 (4) <wsme:NotifyTo> ... </wsme:NotifyTo>
4330 (5) <wsman:ContentEncoding> UTF-16 </wsman:ContentEncoding>
4331 (6) </wsme:Delivery>
4332 (7) </wsme:Subscribe>

```

## 4333 10.2.2 Filtering

4334 Filter expression is constrained to be a Boolean predicate. To support ad hoc queries including  
 4335 projections, WS-Management defines a wsman:Filter element of exactly the same form as what is  
 4336 used in the Subscribe operation except that the filter expression is not constrained to be a Boolean  
 4337 predicate. This allows the use of subscriptions using existing query languages such as SQL and CQL,  
 4338 which combine predicate and projection information in the same syntax. The use of projections is  
 4339 defined by the filter dialect, not by WS-Management.

4340 If the filter dialect for either Filter or wsman:Filter used for the Subscribe message is  
 4341 <http://www.w3.org/TR/1999/REC-xpath-19991116> (the default dialect in both cases), the context node  
 4342 is the SOAP Envelope element.

4343 WS-Management defines the wsman:Filter element as a child of the Subscribe element.

4344 WS-Management defines the wsman:Filter element to allow projections, which is outlined as follows:

```

4345 (1) <wsman:Filter Dialect="xs:anyURI"?> xs:any </wsman:Filter>

```

4346 The Dialect attribute is optional. When not specified, it has the following implied value:

4347 <http://www.w3.org/TR/1999/REC-xpath-19991116>

4348 This dialect allows any full XPath expression or subset to be used.

4349 **R10.2.2-1:** If a service supports filtered subscriptions using Filter, it shall also support filtering  
 4350 using wsman:Filter. This rule allows client stacks to always pick the wsman XML namespace for  
 4351 the Filter element. Even though a service supports wsman:Filter, it is not required to support  
 4352 projections.

4353 **R10.2.2-2:** If a service supports filtered subscriptions using wsman:Filter, it should also support  
 4354 filtering using Filter.

4355 **R10.2.2-3:** If a Subscribe request contains both Filter and wsman:Filter, the service shall return  
 4356 a wsa:InvalidMessage fault.

4357 To allow eventing filter expressions to be defined independently of the delivery mode,  
 4358 WS-Management defines a new filter dialect that is the same as previously defined except that the  
 4359 context node is defined as the element that would be returned as the first child of the SOAP Body  
 4360 element if the Push delivery mode were used. The URI for this filter dialect is:

4361 <http://schemas.dmtf.org/wbem/wsman/1/wsman/filter/eventRootXPath>

4362 The context node for this expression is as follows:

- 4363 • **Context Node:** any XML element that could be returned as a direct child of the s:Body  
 4364 element if the delivery mode was Push
- 4365 • **Context Position:** 1
- 4366 • **Context Size:** 1
- 4367 • **Variable Bindings:** none
- 4368 • **Function Libraries:** Core Function Library [[XPath 1.0](#)]

- 4369       • **Namespace Declarations:** the [in-scope namespaces] property [\[XML Infoset\]](#) of  
4370       /s:Envelope/s:Body/wsme:Subscribe/wsman:Filter

4371       **R10.2.2-4:** Services should support this filter dialect when they want to use an XPath-based  
4372       filter, rather than the default filter dialect defined in 10.2.1.

4373       The considerations described in 8.3 regarding the [XPath 1.0](#) filter dialect also apply to the preceding  
4374       eventing filter.

4375       Resource-constrained implementations might have difficulty providing full XPath processing and yet  
4376       still want to use a subset of XPath syntax. This does not require the addition of a new dialect if the  
4377       expression specified in the filter is a true XPath expression. The use of the filter dialect URI does not  
4378       imply that the service supports the entire specification for that dialect, only that the expression  
4379       conforms to the rules of that dialect. Most services use XPath only for filtering, but they will not  
4380       support the composition of new XML or removing portions of XML that would result in the XML  
4381       fragment violating the schema of the event.

4382       EXAMPLE 1: A typical example of the use of XPath in a subscription follows. Assume that each event that would  
4383       be delivered has the following XML content:

```
4384 (1) <s:Body>
4385 (2) <LowDiskSpaceEvent xmlns="...">
4386 (3) <LogicalDisk>C:</LogicalDisk>
4387 (4) <CurrentMegabytes>12</CurrentMegabytes>
4388 (5) <Megabytes24HoursAgo>17</Megabytes24HoursAgo>
4389 (6) </LowDiskSpaceEvent>
4390 (7) </s:Body>
```

4391       The event is wholly contained within the s:Body of the SOAP message. The anchor point for the  
4392       XPath evaluation is the first element of each event, and it does not reference the <s:Body> element  
4393       as such. The XPath expression is evaluated as if the event content were a separate XML document.

4394       EXAMPLE 2: When used for simple document processing, the following four XPath expressions "select" the  
4395       entire <LowDiskSpaceEvent> node:

```
4396 (8) /
4397 (9) /LowDiskSpaceEvent
4398 (10) ../LowDiskSpaceEvent
4399 (11) .
```

4400       If used as a "filter", this XPath expression does not filter out any instances and is the same as selecting all  
4401       instances of the event, or omitting the filter entirely.

4402       EXAMPLE 3: However, using the following syntax, the XPath expression selects the XML node only if the test  
4403       expression in brackets evaluates to logical "true":

```
4404 (1) ../LowDiskSpaceEvent[LogicalDisk="C:"]
```

4405       In this case, the event is selected if it refers to disk drive "C:"; otherwise the XML node is not selected. This  
4406       XPath expression would filter out all <LowDiskSpaceEvent> events for other drives.

4407       EXAMPLE 4: Full XPath implementations may support more complex test expressions:

```
4408 (1) ../LowDiskSpaceEvent[LogicalDisk="C:" and CurrentMegabytes < "20"]
```

4409       In essence, the XML form of the event is logically passed through the XPath processor to see if it  
4410       would be selected. If so, it is delivered as an event. If not, the event is discarded and not delivered to  
4411       the subscriber.

4412 [XPath 1.0](#) can be used simply for filtering or to send back subsets of the representation (or even the  
4413 values without XML wrappers). In cases where the result is not just filtered but is "altered," the  
4414 technique in 8.6 applies.

4415 If full XPath cannot be supported, a common subset for this purpose is described in ANNEX D of this  
4416 specification.

4417 **R10.2.2-5:** The `wsman:Filter` element shall contain either simple text or a single XML element  
4418 of a single or complex type. A service should reject any filter with mixed content or multiple peer  
4419 XML elements using a `wsme:EventSourceUnableToProcess` fault.

4420 **R10.2.2-6:** A conformant service may not support the entire syntax and processing power of  
4421 the specified filter dialect. The only requirement is that the specified filter is syntactically correct  
4422 within the definition of the dialect. Subsets are therefore legal. If the specified filter exceeds the  
4423 capability of the service, the service should return a `wsman:CannotProcessFilter` fault with text  
4424 explaining why the filter was problematic.

4425 **R10.2.2-7:** If a service requires complex initialization parameters in addition to the filter, these  
4426 should be part of the `wsman:Filter` block because they logically form part of the filter initialization,  
4427 even if some of the parameters are not strictly used in the filtering process. In this case, a unique  
4428 dialect URI shall be devised for the event source and the schema and usage published.

4429 **R10.2.2-8:** If the service supports composition of new XML or filtering to the point where the  
4430 resultant event would not conform to the original schema for that event, the event delivery should  
4431 be wrapped in the same way as content for the fragment-level access operations (see 7.7).

4432 Events, regardless of how they are filtered or reduced, need to conform to some kind of XML schema  
4433 definition when they are actually delivered. Simply sending out unwrapped XML fragments during  
4434 delivery is not legal.

4435 **R10.2.2-9:** If the service requires specific initialization XML in addition to the filter to formulate  
4436 a subscription, this initialization XML shall form part of the filter body and be documented as part  
4437 of the filter dialect.

4438 This rule promotes a consistent location for initialization content, which may be logically seen as part  
4439 of the filter. The filter XML schema is more understandable if it separates the initialization and filtering  
4440 parts into separate XML elements.

4441 For information about filtering over enumerations, see 8.3.

### 4442 10.2.3 Connection Retries

4443 Due to the nature of event delivery, the subscriber might not be reachable at event-time. Rather than  
4444 terminate all subscriptions immediately, typically the service attempts to connect several times with  
4445 suitable timeouts before giving up.

4446 **R10.2.3-1:** A service may observe any connection retry policy or allow the subscriber to define  
4447 it by including the following `wsman:ConnectionRetry` element in a subscription. If the service does  
4448 not accept the `wsman:ConnectionRetry` element, it should return a `wsman:UnsupportedFeature`  
4449 fault with the following detail code:

4450 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DeliveryRetries`

4451 This only applies to failures to *connect* and does not include replay of actual SOAP deliveries.

```
4452 (1) <wsme:Subscribe>
4453 (2) <wsme:Delivery>
4454 (3) <wsme:NotifyTo> ... </wsme:NotifyTo>
```

```

4455 (4) <wsman:ConnectionRetry Total="count"> xs:duration
4456 </wsman:ConnectionRetry>
4457 (5) </wsme:Delivery>
4458 (6) </wsme:Subscribe>

```

4459 The following definitions provide additional, normative constraints on the preceding outline:

4460 wsman:ConnectionRetry

4461 an xs:duration for how long to wait between retries while trying to connect

4462 wsman:ConnectionRetry/@Total

4463 how many retries to attempt, observing the specified interval between the attempts

4464 **R10.2.3-2:** If the retry counts are exhausted, the subscription should be considered abnormally  
4465 terminated.

4466 The retry mechanism applies only to attempts to connect. Failures to deliver on an established  
4467 connection can result in terminating the connection according to the rules of the transport in use, and  
4468 terminating the subscription. Other Web services mechanisms can be used to synthesize reliable  
4469 delivery or safe replay of the actual deliveries.

#### 4470 10.2.4 SubscribeResponse

4471 The service returns any service-specific reference parameters in the SubscriptionManager EPR, and  
4472 these are included by the subscriber (client) later when issuing Unsubscribe and Renew messages.

4473 **R10.2.4-1:** In SubscribeResponse, the service may specify any EPR for the  
4474 SubscriptionManager. However, it is recommended that the address contain the same wsa:To  
4475 address as the original Subscribe request and differ only in other parts of the address, such as  
4476 the reference parameters.

4477 **R10.2.4-2:** A conformant service may not return the Expires field in the response, but, as  
4478 specified in 10.2, this implies that the subscription does not expire until explicitly canceled.

#### 4479 10.2.5 Heartbeats

4480 A typical problem with event subscriptions is a situation in which no event traffic occurs. It is difficult  
4481 for clients to know whether no events matching the subscription have occurred or whether the  
4482 subscription has simply failed and the client was not able to receive any notification.

4483 Because of this, WS-Management defines a "heartbeat" pseudo-event that can be sent periodically  
4484 for any subscription. This event is sent if no regular events occur so that the client knows the  
4485 subscription is still active. If the heartbeat event does not arrive, the client knows that connectivity is  
4486 bad or that the subscription has expired, and it can take corrective action.

4487 The heartbeat event is sent *in place of* the events that would have occurred and is *never* intermixed  
4488 with "real" events. In all modes, including batched, it occurs alone.

4489 To request heartbeat events as part of a subscription, the Subscribe request has an additional field in  
4490 the Delivery section:

```

4491 (1) <wsme:Delivery>
4492 (2) ...
4493 (3) <wsman:Heartbeats> xs:duration </wsman:Heartbeats>
4494 (4) ...
4495 (5) </wsme:Delivery>

```



4496 wsman:Heartbeats specifies that heartbeat events are added to the event stream at the specified  
4497 interval.

4498 **R10.2.5-1:** A service should support heartbeat events. If the service does not support them, it  
4499 shall return a wsman:UnsupportedFeature fault with the following detail code:

4500 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Heartbeats`

4501 Heartbeats apply to all delivery modes.

4502 Heartbeats apply to "pull" mode deliveries as well, in that they are a hint to the publisher about how  
4503 often to expect a Pull request. The service can refuse to deliver events if the client does not regularly  
4504 call back at the heartbeat interval. If no events are available at the heartbeat interval, the service  
4505 simply includes a heartbeat event as the result of the Pull.

4506 **R10.2.5-2:** While a subscription with heartbeats is active, the service shall ensure that either  
4507 real events or heartbeats are sent out within the specified wsman:Heartbeat interval. The service  
4508 may send out heartbeats at this interval in addition to the events, as long as the heartbeat events  
4509 are sent separately (not batched with other events). The goal is to ensure that some kind of event  
4510 traffic always occurs within the heartbeat interval.

4511 **R10.2.5-3:** A conformant service may send out heartbeats at earlier intervals than specified in  
4512 the subscription. However, the events should not be intermixed with other events when batching  
4513 delivery modes are used. Typically, heartbeats are sent out *only when no real events occur*. A  
4514 service may fail to produce heartbeats at the specified interval if real events have been delivered.

4515 **R10.2.5-4:** A conformant service shall not send out heartbeats asynchronously to any event  
4516 deliveries already in progress. They shall be delivered in sequence like any other events,  
4517 although they are delivered alone as single events or as the only event in a batch.

4518 In practice, heartbeat events are based on a countdown timer. If no events occur, the heartbeat is  
4519 sent out alone. However, every time a real event is delivered, the heartbeat countdown timer is reset.  
4520 If a steady stream of events occurs, heartbeats might never be delivered.

4521 Heartbeats need to be acknowledged like any other event if one of the acknowledged delivery modes  
4522 is in effect.

4523 The client assumes that the subscription is no longer active if no heartbeats are received within the  
4524 specified interval, so the service can proceed to cancel the subscription and send any requested  
4525 SubscriptionEnd messages, because the client will likely resubscribe shortly. Used in combination  
4526 with bookmarks (see 10.2.6), heartbeats can achieve highly reliable delivery with known latency  
4527 behavior.

4528 The heartbeat event itself is simply an event message with no body and is identified by its wsa:Action  
4529 URI as follows:

```
4530 (1) <s:Envelope ...>
4531 (2) <s:Header>
4532 (3) <wsa:To> </wsa:To>
4533 (4) <wsa:Action s:mustUnderstand="true">
4534 (5) http://schemas.dmtf.org/wbem/wsman/1/wsman/Heartbeat
4535 (6) </wsa:Action>
4536 (7) ...
4537 (8) </s:Header>
4538 (9) <s:Body/>
4539 (10) </s:Envelope>
```

4540 **10.2.6 Bookmarks**

4541 Reliable delivery of events is difficult to achieve, so management subscribers need to have a way to  
4542 be certain of receiving all events from a source. When subscriptions expire or when deliveries fail,  
4543 windows of time can occur in which the client cannot be certain whether critical events have occurred.  
4544 Rather than using a highly complex, transacted delivery model, WS-Management defines a simple  
4545 mechanism for ensuring that all events are delivered or that dropped events can be detected.

4546 This mechanism requires event sources to be backed by logs, whether short-term or long-term. The  
4547 client subscribes in the same way as a normal Subscribe operation, and specifies that bookmarks are  
4548 to be used. The service then sends a new bookmark with each event delivery, which the client is  
4549 responsible for persisting. This bookmark is essentially a context or a pointer to the logical event  
4550 stream location that matches the subscription filter. As each new delivery occurs, the client updates  
4551 the bookmark in its own space. If the subscription expires or is terminated unexpectedly, the client  
4552 can subscribe again, using the last known bookmark. In essence, the subscription filter identifies the  
4553 desired set of events, and the bookmark tells the service where to start in the log. The client may then  
4554 pick up where it left off.

4555 This mechanism is immune to transaction problems, because the client can simply start from any of  
4556 several recent bookmarks. The only requirement for the service is to have some type of persistent log  
4557 in which to apply the bookmark. If the submitted bookmark is too old (temporally or positionally within  
4558 the log), the service can fault the request, and at least the client reliably knows that events have been  
4559 dropped.

4560 **R10.2.6-1:** A conformant service may support the WS-Management bookmark mechanism. If  
4561 the service does not support bookmarks, it should return a wsman:UnsupportedFeature fault with  
4562 the following detail code:

4563 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Bookmarks`

4564 To request bookmark services, the client includes the wsman:SendBookmarks element in the  
4565 Subscribe request as follows:

```
4566 (1) <s:Body>
4567 (2) <wsme:Subscribe>
4568 (3) <wsme:Delivery>
4569 (4) ...
4570 (5) </wsme:Delivery>
4571 (6) <wsman:SendBookmarks/>
4572 (7) </wsme:Subscribe>
4573 (8) </s:Body>
```

4574 wsman:SendBookmarks instructs the service to send a bookmark with each event delivery.  
4575 Bookmarks apply to all delivery modes.

4576 The bookmark is a token that represents an abstract pointer in the event stream, but whether it points  
4577 to the last delivered event or the last event plus one (the upcoming event) makes no difference  
4578 because the token is supplied to the same implementation during a subsequent Subscribe operation.  
4579 The service can thus attach any service-specific meaning and structure to the bookmark with no  
4580 change to the client.

4581 If bookmarks are requested, each event delivery contains a new bookmark value as a SOAP header,  
4582 as shown in the following outline. The format of the bookmark is entirely determined by the service  
4583 and is treated as an opaque value by the client.

```
4584 (1) <s:Envelope
4585 (2) xmlns:s="http://www.w3.org/2003/05/soap-envelope"
4586 (3) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
4587 (4) xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
```

```

4588 (5) <s:Header>
4589 (6) <wsa:To s:mustUnderstand="true">http://2.3.4.5/client</wsa:To>
4590 (7) ...
4591 (8) <wsman:Bookmark> xs:any </wsman:Bookmark>
4592 (9) ...
4593 (10) </s:Header>
4594 (11) <s:Body>
4595 (12) ...event content...
4596 (13) </s:Body>
4597 (14) </s:Envelope>

```

4598 wsman:Bookmark contains XML content supplied by the service that indicates the logical position of  
4599 this event or event batch in the event stream implied by the subscription.

4600 **R10.2.6-2:** If bookmarks are supported, the wsman:Bookmark element content shall be either  
4601 simple text or a single complex XML element. A conformant service shall not accept mixed  
4602 content of both text and elements, or multiple peer XML elements, under the wsman:Bookmark  
4603 element.

4604 **R10.2.6-3:** If bookmarks are supported, the service shall use a wsman:Bookmark element in  
4605 the header to send an updated bookmark with each event delivery. Bookmarks accompany only  
4606 event deliveries and are not part of any SubscriptionEnd message.

4607 After the subscription has terminated, for whatever reason, a subsequent Subscribe message on the  
4608 part of the client can include the bookmark in the subscription request. The service then knows where  
4609 to start.

4610 The last-known bookmark received by the client is added to the Subscribe message as a new block,  
4611 positioned after the child elements of Subscribe, as in the following outline:

```

4612 (1) <s:Body>
4613 (2) <wsme:Subscribe>
4614 (3) <wsme:Delivery> ... </wsme:Delivery>
4615 (4) <wsme:Expires> ... </wsme:Expires>
4616 (5) <wsman:Filter> ... </wsman:Filter>
4617 (6) <wsman:Bookmark>
4618 (7) ...last known bookmark from a previous delivery...
4619 (8) </wsman:Bookmark>
4620 (9) <wsman:SendBookmarks/>
4621 (10) </wsme:Subscribe>
4622 (11) </s:Body>

```

4623 The following definitions provide additional, normative constraints on the preceding outline:

4624 wsman:Bookmark  
4625 arbitrary XML content previously supplied by the service as a wsman:Bookmark during event  
4626 deliveries from a previous subscription

4627 wsman:SendBookmarks  
4628 an instruction to continue delivering updated bookmarks with each event delivery

4629 **R10.2.6-4:** The bookmark is a pointer to the last event delivery or batched delivery. The service  
4630 shall resume delivery at the first event or events after the event represented by the bookmark.  
4631 The service shall not replay events associated with the bookmark or skip any events since the  
4632 bookmark.

4633 **R10.2.6-5:** The service may support a short queue of previous bookmarks, allowing the  
4634 subscriber to start using any of several previous bookmarks. If bookmarks are supported, the

4635 service is required only to support the most recent bookmark for which delivery had apparently  
4636 succeeded.

4637 **R10.2.6-6:** If the bookmark cannot be honored, the service shall fault with a  
4638 wsman:InvalidBookmark fault with one of the following detail codes:

- 4639 • bookmark has expired (the source is not able to back up and replay from that point):  
4640 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Expired>
- 4641 • format is unknown:  
4642 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFormat>

4643 If multiple new subscriptions are made using a previous bookmark, the service can allow multiple  
4644 reuse or may limit bookmarks to a single subscriber, and can even restrict how long bookmarks can  
4645 be used before becoming invalid.

4646 The following predefined, reserved bookmark value indicates that the subscription starts at the  
4647 earliest possible point in the event stream backed by the publisher:

4648 <http://schemas.dmtf.org/wbem/wsman/1/wsman/bookmark/earliest>

4649 If a subscription is received with this bookmark, the event source replays all possible events that  
4650 match the filter and any events that subsequently occur for that event source. The absence of any  
4651 bookmark means "begin at the next available event".

4652 **R10.2.6-7:** A conformant service may support the reserved bookmark  
4653 <http://schemas.dmtf.org/wbem/wsman/1/wsman/bookmark/earliest> and not support any other type  
4654 of bookmark. If the <http://schemas.dmtf.org/wbem/wsman/1/wsman/bookmark/earliest> bookmark  
4655 is supported, the event source should send all previous and future events that match the filter  
4656 starting with the earliest such event.

## 4657 **10.2.7 Delivery Modes**

4658 While the general pattern of asynchronous, event-based messages is extremely common, different  
4659 applications often require different event message delivery mechanisms. For instance, in some cases  
4660 a simple asynchronous message is optimal, while other situations may work better if the event  
4661 consumer can poll for event messages in order to control the flow and timing of message arrival.  
4662 Some consumers require event messages to be wrapped in a standard "event" SOAP envelope,  
4663 while others prefer messages to be delivered unwrapped. Some consumers may require event  
4664 messages to be delivered reliably, while others may be willing to accept best-effort event delivery.

4665 In order to support this broad variety of event delivery requirements, this specification introduces an  
4666 abstraction called a Delivery Mode. This concept is used as an extension point, so that event sources  
4667 and event consumers may freely create new delivery mechanisms that are tailored to their specific  
4668 requirements. This specification provides a minimal amount of support for delivery mode negotiation  
4669 by allowing an event source to provide a list of supported delivery modes in response to a  
4670 subscription request specifying a delivery mode it does not support.

4671 A WS-Management implementation can support a variety of event delivery modes.

4672 In essence, delivery consists of the following items:

- 4673 • a delivery mode (how events are packaged)
- 4674 • an address (the transport and network location)
- 4675 • an authentication profile to use when connecting or delivering the events (security)

4676 The standard security profiles are discussed in clause 12 and may be required for subscriptions if the  
4677 service needs hints or other indications of which security model to use at event-time.

4678 If the delivery mode is supported but not actually usable due to firewall configuration, the service can  
4679 return a wsme:DeliveryModeRequestedUnavailable fault with additional detail to this effect.

4680 **R10.2.7-1:** For any given transport, a conformant service should support at least one of the  
4681 following delivery modes to interoperate with standard clients:

4682 <http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push>

4683 <http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck>

4684 <http://schemas.dmtf.org/wbem/wsman/1/wsman/Events>

4685 <http://schemas.dmtf.org/wbem/wsman/1/wsman/Pull>

4686 The delivery mode does *not* imply any specific transport.

4687 Modes describe SOAP message behavior and are unrelated to the transport that is in use. A delivery  
4688 mode implies a specific SOAP message format, so a message that deviates from that format requires  
4689 a new delivery mode.

4690 **R10.2.7-2:** The NotifyTo address in the Subscribe message shall support only a single delivery  
4691 mode.

4692 This requirement is for the client because the service cannot verify whether this statement is true. If  
4693 this requirement is not observed by the client, the service might not operate correctly. If the  
4694 subscriber supports multiple delivery modes, the NotifyTo address needs to be differentiated in some  
4695 way, such as by adding an additional reference parameter.

## 4696 10.2.8 Event Action URI

4697 Typically, each event type has its own wsa:Action URI to quickly identify and route the event. If an  
4698 event type does not define its own wsa:Action URI, the following URI can be used as a default:

4699 <http://schemas.dmtf.org/wbem/wsman/1/wsman/Event>

4700 This URI can be used in cases where event types are inferred in real-time from other sources and not  
4701 published as Web service events, and thus do not have a designated wsa:Action URI. This  
4702 specification places no restrictions on the wsa:Action URI for events. More specific URIs can act as a  
4703 reliable dispatching point. In many cases, a fixed schema can serve to model many different types of  
4704 events, in which case the event "ID" is simply a field in the XML content of the event. The URI in this  
4705 case might reflect the schema and be undifferentiated for all of the various event IDs that might occur  
4706 or it might reflect the specific event by suffixing the event ID to the wsa:Action URI. This specification  
4707 places no restrictions on the granularity of the URI, but careful consideration of these issues is part of  
4708 designing the URIs for events.

## 4709 10.2.9 Delivery Sequencing and Acknowledgement

4710 The delivery mode indicates how the service will exchange events with interested parties. This clause  
4711 describes delivery modes in detail.

4712 **10.2.9.1 General**

4713 For some event types, ordered and acknowledged delivery is important, but for other types of events  
4714 the order of arrival is not significant. WS-Management defines four standard delivery modes:

- 4715 • <http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push>  
4716 With this mode, each SOAP message has only one event and no acknowledgement or  
4717 SOAP response. The service can deliver events for the subscription asynchronously without  
4718 regard to any events already in transit. This mode is useful when the order of events does  
4719 not matter, such as with events containing running totals in which each new event can  
4720 replace the previous one completely and the time stamp is sufficient for identifying the most  
4721 recent event.
- 4722 • <http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck>  
4723 With this mode, each SOAP message has only one event, but each event is acknowledged  
4724 before another is sent. The service queues all undelivered events for the subscription and  
4725 delivers each new event only after the previous one has been acknowledged.
- 4726 • <http://schemas.dmtf.org/wbem/wsman/1/wsman/Events>  
4727 With this mode, each SOAP message can have many events, but each batch is  
4728 acknowledged before another is sent. The service queues all events for the subscription  
4729 and delivers them in that order, maintaining the order in the batches.
- 4730 • <http://schemas.dmtf.org/wbem/wsman/1/wsman/Pull>  
4731 With this mode, each SOAP message can have many events, but each batch is  
4732 acknowledged. Because the receiver uses Pull to synchronously retrieve the events,  
4733 acknowledgement is implicit. The order of delivery is maintained.

4734 Ordering of events across subscriptions is not implied.

4735 The acknowledgement model is discussed in 10.8.

4736 **10.2.9.2 Push Mode**

4737 The standard delivery mode is  
4738 <http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push>, in which each delivery  
4739 consists of a single event. No acknowledgement occurs, so the delivery cannot be faulted to cancel  
4740 the subscription.

4741 Therefore, subscriptions made with this delivery mode can have short durations to prevent a situation  
4742 in which deliveries cannot be stopped if the SubscriptionManager content from the  
4743 SubscribeResponse information is corrupted or lost.

4744 To promote fast routing of events, the required wsa:Action URI in each event message can be distinct  
4745 for each event type, regardless of how strongly typed the event body is.

4746 **R10.2.9.2-1:** A service may support the  
4747 <http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push> delivery mode.

4748 **R10.2.9.2-2:** To precisely control how to deal with events that are too large, the service may  
4749 accept the following additional instruction in a subscription:

```
4750 (1) <wsme:Delivery>
4751 (2) <wsme:NotifyTo> ... </wsme:NotifyTo>
4752 (3) ...
4753 (4) <wsman:MaxEnvelopeSize Policy="enumConstant">
```

```

4754 (5) xs:positiveInteger
4755 (6) </wsman:MaxEnvelopeSize>
4756 (7) ...
4757 (8) </wsme:Delivery>

```

4758 The following definitions provide additional, normative constraints on the preceding outline:

4759 wsme:Delivery/wsman:MaxEnvelopeSize

4760 the maximum number of octets for the entire SOAP envelope in a single event delivery

4761 wsme:Delivery/wsman:MaxEnvelopeSize/@Policy

4762 an optional value with one of the following enumeration values:

- 4763 • **CancelSubscription:** cancel on the first oversized event
- 4764 • **Skip:** silently skip oversized events
- 4765 • **Notify:** notify the subscriber that events were dropped as specified in 10.9

4766 **R10.2.9.2-3:** If wsman:MaxEnvelopeSize is requested, the service shall not send an event  
 4767 body larger than the specified limit. The default behavior is to notify the subscriber as specified in  
 4768 10.9, unless otherwise instructed in the subscription, and to attempt to continue delivery. If the  
 4769 event exceeds any internal default maximums, the service should also attempt to notify as  
 4770 specified in 10.9 rather than terminate the subscription, unless otherwise specified in the  
 4771 subscription. If wsman:MaxEnvelopeSize is too large for the service, the service shall return a  
 4772 wsman:EncodingLimit fault with the following detail code:

4773 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize>

4774 In the absence of any other Policy instructions, services are to deliver notifications of dropped events  
 4775 to subscribers, as specified in 10.9.

### 4776 10.2.9.3 PushWithAck Mode

4777 This delivery mode is identical to the standard "Push" mode except that each delivery is  
 4778 acknowledged. Each delivery still has one event, and the wsa:Action element indicates the event  
 4779 type. However, a SOAP-based acknowledgement occurs as described in 10.7.

4780 The delivery mode URI is:

4781 <http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck>

4782 In every other respect except the delivery mode URI, this mode is identical to Push mode as  
 4783 described in 10.2.9.2.

4784 **R10.2.9.3-1:** A service should support the  
 4785 <http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck> delivery mode. If the delivery mode  
 4786 is not supported, the service should return a fault of wsme:DeliveryModeRequestedUnavailable.

### 4787 10.2.9.4 Batched Delivery Mode

4788 Batching events is an effective way to minimize event traffic from a high-volume event source without  
 4789 sacrificing event timeliness. WS-Management defines a custom event delivery mode that allows an  
 4790 event source to bundle multiple outgoing event messages into a single SOAP envelope. Delivery is  
 4791 always acknowledged, using the model defined in 10.7.

4792 **R10.2.9.4-1:** A service may support the <http://schemas.dmtf.org/wbem/wsman/1/wsman/Events>  
 4793 delivery mode. If the delivery mode is not supported, the service should return a fault of  
 4794 wsme:DeliveryModeRequestedUnavailable.

4795 For this delivery mode, the Delivery element has the following format:

```

4796 (1) <wsme:Delivery Mode="http://schemas.dmtf.org/wbem/wsman/1/wsman/Events">
4797 (2) <wsme:NotifyTo>
4798 (3) wsa:EndpointReferenceType
4799 (4) </wsme:NotifyTo>
4800 (5) <wsman:MaxElements> xs:positiveInteger </wsman:MaxElements> ?
4801 (6) <wsman:MaxTime> xs:duration </wsman:MaxTime> ?
4802 (7) <wsman:MaxEnvelopeSize Policy="enumConstant">
4803 (8) xs:positiveInteger
4804 (9) </wsman:MaxEnvelopeSize> ?
4805 (10) </wsme:Delivery>

```

4806 The following definitions provide additional, normative constraints on the preceding outline:

4807 wsme:Delivery/@Mode

4808 required attribute that shall be defined as

4809 <http://schemas.dmtf.org/wbem/wsman/1/wsman/Events>

4810 wsme:Delivery/wsme:NotifyTo

4811 required element that shall contain the EPR to which event messages are to be sent for this  
4812 subscription

4813 wsme:Delivery/wsman:MaxElements

4814 optional element that contains a positive integer that indicates the maximum number of event  
4815 bodies to batch into a single SOAP envelope

4816 The resource shall not deliver more than this number of items in a single delivery, although it  
4817 may deliver fewer.

4818 wsme:Delivery/wsman:MaxEnvelopeSize

4819 optional element that contains a positive integer that indicates the maximum number of octets in  
4820 the SOAP envelope used to deliver the events

4821 wsman:MaxEnvelopeSize/@Policy

4822 an optional attribute with one of the following enumeration values:

- 4823 • **CancelSubscription:** cancel on the first oversized event
- 4824 • **Skip:** silently skip oversized events
- 4825 • **Notify:** notify the subscriber that events were dropped as specified in 10.9

4826 wsme:Delivery/wsman:MaxTime

4827 optional element that contains a duration that indicates the maximum amount of time the service  
4828 should allow to elapse while batching Event bodies

4829 This time may not be exceeded between the encoding of the first event in the batch and the  
4830 dispatching of the batch for delivery. Some publisher implementations may choose more  
4831 complex schemes in which different events included in the subscription are delivered at different  
4832 latencies or at different priorities. In such cases, a specific filter dialect can be designed for the  
4833 purpose and used to describe the instructions to the publisher. In such cases, wsman:MaxTime  
4834 can be omitted if it is not applicable; if present, however, it serves as an override of anything  
4835 defined within the filter.



4836 In the absence of any other instructions in any part of the subscription, services are to deliver  
4837 notifications of dropped events to subscribers, as specified in 10.9.

4838 If a client wants to discover the appropriate values for wsman:MaxElements or  
4839 wsman:MaxEnvelopeSize, the client can query for service-specific metadata. The format of such  
4840 metadata is beyond the scope of this particular specification.

4841 **R10.2.9.4-2:** If batched mode is requested in a Subscribe message, and MaxElements,  
4842 MaxEnvelopeSize, and MaxTime elements are not present, the service may pick any applicable  
4843 defaults. The following faults apply:

4844 • If MaxElements is not supported, wsman:UnsupportedFeature is returned with the following  
4845 fault detail code:

4846 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxElements`

4847 • If MaxEnvelopeSize is not supported, wsman:UnsupportedFeature is returned with the  
4848 following fault detail code:

4849 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize`

4850 • If MaxTime is not supported, wsman:UnsupportedFeature is returned with the following fault  
4851 detail code:

4852 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxTime`

4853 • If MaxEnvelopeSize/@Policy is not supported, wsman:UnsupportedFeature is returned with  
4854 the following fault detail code:

4855 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopePolicy`

4856 **R10.2.9.4-3:** If wsman:MaxEnvelopeSize is requested, the service shall not send an event  
4857 body larger than the specified limit. The default behavior is to notify the subscriber as specified in  
4858 10.9, unless otherwise instructed in the subscription, and to attempt to continue delivery. If the  
4859 event exceeds any internal default maximums, the service should also attempt notification as  
4860 specified in 10.9 rather than terminate the subscription, unless otherwise specified in the  
4861 subscription.

4862 If a subscription has been created using batched mode, all event delivery messages shall have  
4863 the following format:

```
4864 (1) <s:Envelope ...>
4865 (2) <s:Header>
4866 (3) ...
4867 (4) <wsa:Action>
4868 (5) http://schemas.dmtf.org/wbem/wsman/1/wsman/Events
4869 (6) </wsa:Action>
4870 (7) ...
4871 (8) </s:Header>
4872 (9) <s:Body>
4873 (10) <wsman:Events>
4874 (11) <wsman:Event Action="event action URI">
4875 (12) ...event body...
4876 (13) </wsman:Event> +
4877 (14) </wsman:Events>
4878 (15) </s:Body>
4879 (16) </s:Envelope>
```

4880 The following definitions provide additional, normative constraints on the preceding outline:

4881 s:Envelope/s:Header/wsa:Action

4882 required element that shall be defined as

4883 `http://schemas.dmtf.org/wbem/wsman/1/wsman/Events`

4884 s:Envelope/s:Body/wsman:Events/wsman:Event

4885 required elements that shall contain the body of the corresponding event message, as if  
4886 wsman:Event were the s:Body element

4887 s:Envelope/s:Body/wsman:Events/wsman:Event/@Action

4888 required attribute that shall contain the wsa:Action URI that would have been used for the  
4889 contained event message

4890 **R10.2.9.4-4:** If batched mode is requested, deliveries shall be acknowledged as described in  
4891 10.7.

4892 Dropped events (as specified in 10.9) are encoded with any other events.

4893 **EXAMPLE:** The following example shows batching parameters supplied to a Subscribe operation. The  
4894 service is instructed to send no more than 10 items per batch, to wait no more than 20 seconds from the  
4895 time the first event is encoded until the entire batch is dispatched, and to include no more than 8192 octets  
4896 in the SOAP message.

```
4897 (1) ...
4898 (2) <wsme:Delivery
4899 (3) Mode="http://schemas.dmtf.org/wbem/wsman/1/wsman/Events">
4900 (4) <wsme:NotifyTo>
4901 (5) <wsa:Address>http://2.3.4.5/client</wsa:Address>
4902 (6) </wsme:NotifyTo>
4903 (7) <wsman:MaxElements>10</wsman:MaxElements>
4904 (8) <wsman:MaxTime>PT20S</wsman:MaxTime>
4905 (9) <wsman:MaxEnvelopeSize>8192</wsman:MaxEnvelopeSize>
4906 (10) </wsme:Delivery>
```

4907 **EXAMPLE:** Following is an example of batched delivery that conforms to this specification:

```
4908 (1) <s:Envelope
4909 (2) xmlns:s="http://www.w3.org/2003/05/soap-envelope"
4910 (3) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
4911 (4) xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
4912 (5) xmlns:wsme="http://schemas.xmlsoap.org/ws/2004/08/eventing">
4913 (6) <s:Header>
4914 (7) <wsa:To s:mustUnderstand="true">http://2.3.4.5/client</wsa:To>
4915 (8) <wsa:Action>
4916 (9) http://schemas.dmtf.org/wbem/wsman/1/wsman/Events
4917 (10) </wsa:Action>
4918 (11) ...
4919 (12) </s:Header>
4920 (13) <s:Body>
4921 (14) <wsman:Events>
4922 (15) <wsman:Event
4923 (16) Action="http://schemas.xmlsoap.org/2005/02/diskspacechange">
4924 (17) <DiskChange
4925 (18) xmlns="http://schemas.xmlsoap.org/2005/02/diskspacechange">
4926 (19) <Drive> C: </Drive>
```

```

4927 (20) <FreeSpace> 802012911 </FreeSpace>
4928 (21) </DiskChange>
4929 (22) </wsman:Event>
4930 (23) <wsman:Event
4931 (24) Action="http://schemas.xmlsoap.org/2005/02/diskspacechange">
4932 (25) <DiskChange
4933 (26) xmlns="http://schemas.xmlsoap.org/2005/02/diskspacechange">
4934 (27) <Drive> D: </Drive>
4935 (28) <FreeSpace> 1402012913 </FreeSpace>
4936 (29) </DiskChange>
4937 (30) </wsman:Event>
4938 (31) </wsman:Events>
4939 (32) </s:Body>
4940 (33) </s:Envelope>

```

4941 The Action URI in line 9 specifies that this is a batch that contains distinct events. The individual  
 4942 event bodies are at lines 15–22 and lines 23–30. The actual Action attribute for the individual events  
 4943 is an attribute of the wsman:Event wrapper.

#### 4944 10.2.9.5 Pull Delivery Mode

4945 In some circumstances, polling for events is an effective way of controlling data flow and balancing  
 4946 timeliness against processing ability. Also, in some cases, network restrictions prevent "push" modes  
 4947 from being used; that is, the service cannot initiate a connection to the subscriber.

4948 WS-Management defines a custom event delivery mode, "pull mode," which allows an event source  
 4949 to maintain a logical queue of event messages received by enumeration. This delivery mode borrows  
 4950 the Pull message to retrieve events from the logical queue. However, all of the other pub/sub  
 4951 operations defined in this clause can continue to be used. (For example, Unsubscribe, rather than  
 4952 Release, is used to cancel a subscription.)

4953 For this delivery mode, the Delivery element has the following format:

```

4954 (1) <wsme:Delivery Mode="http://schemas.dmtf.org/wbem/wsman/1/wsman/Pull">
4955 (2) ...
4956 (3) </wsme:Delivery>

```

4957 wsme:Delivery/@Mode shall be

4958 <http://schemas.dmtf.org/wbem/wsman/1/wsman/Pull>

4959 **R10.2.9.5-1:** A service may support the <http://schemas.dmtf.org/wbem/wsman/1/wsman/Pull>  
 4960 delivery mode. If pull mode is requested but not supported, the service shall return a fault of  
 4961 wsme:DeliveryModeRequestedUnavailable.

4962 wsman:MaxElements, wsman:MaxEnvelopeSize, and wsman:MaxTime do not apply in the Subscribe  
 4963 message when using this delivery mode because the Pull message contains all of the necessary  
 4964 functionality for controlling the batching and timing of the responses.

4965 **R10.2.9.5-2:** If a subscription incorrectly specifies parameters that are not compatible with pull  
 4966 mode, the service should issue a wsman:UnsupportedFeature fault with the following detail code:

4967 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FormatMismatch>

4968 **R10.2.9.5-3:** If pull mode is requested in a Subscribe message and the event source accepts  
 4969 the subscription request, the SubscribeResponse element in the REPLY message shall contain  
 4970 an EnumerationContext element suitable for use in a subsequent Pull operation.

4971 EXAMPLE:

```

4972 (1) <s:Body ...>
4973 (2) <wsme:SubscribeResponse ...>
4974 (3) <wsme:SubscriptionManager>
4975 (4) wsa:EndpointReferenceType
4976 (5) </wsme:SubscriptionManager>
4977 (6) <wsme:Expires>[xs:dateTime | xs:duration]</wsme:Expires>
4978 (7) <wsmen:EnumerationContext>...</wsmen:EnumerationContext>
4979 (8) ...
4980 (9) </wsme:SubscribeResponse>
4981 (10) </s:Body>

```

4982 The subscriber extracts the EnumerationContext and uses it thereafter in Pull requests.

4983 **R10.2.9.5-4:** If pull mode is active, Pull messages shall use the EPR of the subscription  
 4984 manager obtained from the SubscribeResponse message. The EPR reference parameters are of  
 4985 a service-specific addressing model, but may use the WS-Management default addressing model  
 4986 if it is suitable.

4987 **R10.2.9.5-5:** If pull mode is active and a Pull request returns no events (because none have  
 4988 occurred since the last "pull"), the service should return a wsman:TimedOut fault. The  
 4989 EnumerationContext is still considered active, and the subscriber may continue to issue Pull  
 4990 requests with the most recent EnumerationContext for which event deliveries actually occurred.

4991 **R10.2.9.5-6:** If pull mode is active and a Pull request returns events, the service may return an  
 4992 updated EnumerationContext as specified for Pull, and the subscriber is expected to use the  
 4993 update, if any, in the subsequent Pull, as specified for the Enumeration operations. Bookmarks, if  
 4994 active, may also be returned in the header and shall also be updated by the service.

4995 In practice, the service might not actually change the EnumerationContext, but the client cannot  
 4996 depend on it remaining constant. It is updated conceptually, if not actually.

4997 In pull mode, the Pull request controls the batching. If no defaults are specified, the batch size is 1  
 4998 and the maximum envelope size and timeouts are service-defined.

4999 **R10.2.9.5-7:** If pull mode is active, the service shall not return an EndOfSequence element in  
 5000 the event stream because no concept of a "last event" exists in this mode. Rather, the  
 5001 enumeration context should become invalid if the subscription expires or is canceled for any  
 5002 reason.

5003 **R10.2.9.5-8:** If pull mode is used, the service shall accept the wsman:MaxEnvelopeSize used  
 5004 in the Pull as the limitation on the event size that can be delivered.

5005 The batching properties used in batched mode do not apply to pull mode. The client controls the  
 5006 maximum event size using the normal mechanisms in Pull.

### 5007 10.3 GetStatus

5008 To get the status of a subscription, the subscriber sends a request of the following form to the  
 5009 subscription manager:

```

5010 (1) <s:Envelope ...>
5011 (2) <s:Header ...>
5012 (3) <wsa:Action>
5013 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatus
5014 (5) </wsa:Action>
5015 (6) ...
5016 (7) </s:Header>

```

```

5017 (8) <s:Body ...>
5018 (9) <wsme:GetStatus ...>
5019 (10) ...
5020 (11) </wsme:GetStatus>
5021 (12) </s:Body>
5022 (13) </s:Envelope>

```

5023 Components of the preceding outline are additionally constrained as for a request to renew a  
5024 subscription. Other components of the preceding outline are not further constrained by this  
5025 specification.

5026 If the subscription is valid and has not expired, the subscription manager shall reply with a response  
5027 of the following form:

```

5028 (1) <s:Envelope ...>
5029 (2) <s:Header ...>
5030 (3) <wsa:Action>
5031 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatusResponse
5032 (5) </wsa:Action>
5033 (6) ...
5034 (7) </s:Header>
5035 (8) <s:Body ...>
5036 (9) <wsme:GetStatusResponse ...>
5037 (10) <wsme:Expires>[xs:dateTime | xs:duration]</wsme:Expires> ?
5038 (11) ...
5039 (12) </wsme:GetStatusResponse>
5040 (13) </s:Body>
5041 (14) </s:Envelope>

```

5042 Components of the preceding outline are constrained as for a response to a renew request. Other  
5043 components of the preceding outline are not further constrained by this specification.

5044 The wsme:GetStatus message is optional for WS-Management.

5045 **R10.3-1:** The wse:GetStatus message in a constrained environment is a candidate for exclusion.  
5046 If this message is not supported, then a wsa:ActionNotSupported fault shall be returned in  
5047 response to this request.

5048 Heartbeat support may be implemented rather than the wsme:GetStatus message.

## 5049 10.4 Unsubscribe

5050 Though subscriptions expire eventually, to minimize resources the subscribing event sink should  
5051 explicitly delete a subscription when it no longer wants notifications associated with the subscription.

5052 To explicitly delete a subscription, a subscribing event sink sends a request of the following form to  
5053 the subscription manager:

```

5054 (1) <s:Envelope ...>
5055 (2) <s:Header ...>
5056 (3) <wsa:Action>
5057 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/Unsubscribe
5058 (5) </wsa:Action>
5059 (6) ...
5060 (7) </s:Header>
5061 (8) <s:Body>
5062 (9) <wsme:Unsubscribe ...>
5063 (10) ...
5064 (11) </wsme:Unsubscribe>
5065 (12) </s:Body>
5066 (13) </s:Envelope>

```

5067 Components of the preceding outline are additionally constrained only as for a request to renew a  
 5068 subscription. For example, the faults listed there are also defined for a request to delete a  
 5069 subscription.

5070 If the subscription manager accepts a request to delete a subscription, it shall reply with a response  
 5071 of the following form:

```
5072 (1) <s:Envelope ...>
5073 (2) <s:Header ...>
5074 (3) <wsa:Action>
5075 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/UnsubscribeResponse
5076 (5) </wsa:Action>
5077 (6) <wsa:RelatesTo>xs:anyURI</wsa:RelatesTo>
5078 (7) ...
5079 (8) </s:Header>
5080 (9) <s:Body />
5081 (10) </s:Envelope>
```

5082 Components of the preceding outline are not further constrained by this specification.

5083 **R10.4-1:** If a service supports Subscribe, it shall implement the Unsubscribe message and  
 5084 ensure that event delivery will be terminated if the message is accepted as valid. Delivery of  
 5085 events may occur after responding to the Unsubscribe message as long as the event traffic stops  
 5086 at some point.

5087 **R10.4-2:** A service may unilaterally cancel a subscription for any reason, including internal  
 5088 timeouts, reconfiguration, or unreliable connectivity.

5089 Clients need to be prepared to receive any events already in transit even though they have issued an  
 5090 Unsubscribe message. Clients have the option to either fault any such deliveries or accept them.

5091 The EPR to use for this message is received from the SubscribeResponse element in the  
 5092 SubscriptionManager element.

## 5093 10.5 Renew

5094 To update the expiration for a subscription, subscription managers shall support requests to renew  
 5095 subscriptions.

5096 To renew a subscription, the subscriber sends a request of the following form to the subscription  
 5097 manager:

```
5098 (1) <s:Envelope ...>
5099 (2) <s:Header ...>
5100 (3) <wsa:Action>
5101 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/Renew
5102 (5) </wsa:Action>
5103 (6) ...
5104 (7) </s:Header>
5105 (8) <s:Body ...>
5106 (9) <wsme:Renew ...>
5107 (10) <wsme:Expires>[xs:dateTime | xs:duration]</wsme:Expires> ?
5108 (11) ...
5109 (12) </wsme:Renew>
5110 (13) </s:Body>
5111 (14) </s:Envelope>
```

5112 Components of the preceding outline are additionally constrained as for a request to create a  
 5113 subscription. Other components of the preceding outline are not further constrained by this  
 5114 specification.

5115 If the subscription manager accepts a request to renew a subscription, it shall reply with a response  
5116 of the following form:

```
5117 (1) <s:Envelope ...>
5118 (2) <s:Header ...>
5119 (3) <wsa:Action>
5120 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/RenewResponse
5121 (5) </wsa:Action>
5122 (6) ...
5123 (7) </s:Header>
5124 (8) <s:Body ...>
5125 (9) <wsme:RenewResponse ...>
5126 (10) <wsme:Expires>[xs:dateTime | xs:duration]</wsme:Expires> ?
5127 (11) ...
5128 (12) </wsme:RenewResponse>
5129 (13) </s:Body>
5130 (14) </s:Envelope>
```

5131 Components of the preceding outline are constrained as for a response to a subscribe request with  
5132 the following addition(s):

5133 /s:Envelope/s:Body\*/wsme:Expires

5134 If the requested expiration is a duration, then the implied start of that duration is the time when  
5135 the subscription manager starts processing the Renew request.

5136 If the subscription manager chooses not to renew this subscription, the request shall fail, and the  
5137 subscription manager may generate a wsme:UnableToRenew fault indicating that the renewal was  
5138 not accepted.

5139 Other components of the preceding outline are not further constrained by this specification.

5140 Processing of the Renew message is required, but it is not required to succeed.

5141 **R10.5-1:** Although a conformant service shall accept the Renew message as a valid action, the  
5142 service may always fault the request with a wsme:UnableToRenew fault, forcing the client to  
5143 subscribe from scratch.

5144 Renew has no effect on deliveries in progress, bookmarks, heartbeats, or other ongoing activity. It  
5145 simply extends the lifetime of the subscription.

5146 The EPR to use for this message is received from the SubscribeResponse element in the  
5147 SubscriptionManager element.

## 5148 10.6 SubscriptionEnd

5149 If the event source terminates a subscription unexpectedly, the event source should send a  
5150 Subscription End SOAP message to the endpoint reference indicated when the subscription was  
5151 created. The message shall be of the following form:

```
5152 (1) <s:Envelope ...>
5153 (2) <s:Header ...>
5154 (3) <wsa:Action>
5155 (4) http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscriptionEnd
5156 (5) </wsa:Action> ?
5157 (6) ...
5158 (7) </s:Header>
5159 (8) <s:Body ...>
5160 (9) <wsme:SubscriptionEnd ...>
5161 (10) <wsme:SubscriptionManager>
5162 (11) endpoint-reference
```

```

5163 (12) </wsme:SubscriptionManager>
5164 (13) <wsme:Status>
5165 (14) [
5166 (15) http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryFailure |
5167 (16) http://schemas.xmlsoap.org/ws/2004/08/eventing/SourceShuttingDown |
5168 (17) http://schemas.xmlsoap.org/ws/2004/08/eventing/SourceCancelling
5169 (18)]
5170 (19) </wsme:Status>
5171 (20) <wsme:Reason xml:lang="language identifier" >xs:string</wsme:Reason>
5172 ?
5173 (21) ...
5174 (22) </wsme:SubscriptionEnd>
5175 (23) ...
5176 (24) </s:Body>
5177 (25) </s:Envelope>

```

5178 The following describes additional, normative constraints on the preceding outline:

5179 `/s:Envelope/s:Body*/wsme:SubscriptionManager`

5180 Endpoint reference of the subscription manager. It is recommended that event sinks ignore this  
5181 element as its usage requires the ability to compare EPRs for equality when no such mechanism  
5182 exists. Event sinks are advised to use reference parameters in the  
5183 `/wsme:Subscribe/wsme:EndTo` EPR if they wish to correlate this message against their  
5184 outstanding subscriptions.

5185 `/s:Envelope/s:Body/wsme:SubscriptionEnd/wsme:Status =`

5186 `"http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryFailure"`

5187 This value shall be used if the event source terminated the subscription because of problems  
5188 delivering notifications.

5189 `/s:Envelope/s:Body/wsme:SubscriptionEnd/wsme:Status =`

5190 `"http://schemas.xmlsoap.org/ws/2004/08/eventing/SourceShuttingDown"`

5191 This value shall be used if the event source terminated the subscription because the source is  
5192 being shut down in a controlled manner (that is, if the event source is being shut down but has  
5193 the opportunity to send a `SubscriptionEnd` message before it exits).

5194 `/s:Envelope/s:Body/wsme:SubscriptionEnd/wsme:Status =`

5195 `"http://schemas.xmlsoap.org/ws/2004/08/eventing/SourceCancelling"`

5196 This value shall be used if the event source terminated the subscription for some other reason  
5197 before it expired.

5198 `/s:Envelope/s:Body/wsme:SubscriptionEnd/wsme:Reason`

5199 This optional element contains text, in the language specified by the `@xml:lang` attribute,  
5200 describing the reason for the unexpected subscription termination.

5201 Other message information headers defined in 5.4 may be included in the message, according to the  
5202 usage and semantics defined in 5.4.

5203 Other components of the preceding outline are not further constrained by this specification.

5204 This `SubscriptionEnd` message is optional for WS-Management. In effect, it is the "last event" for a  
5205 subscription. Because its primary purpose is to warn a subscriber that a subscription has ended, it is  
5206 not suitable for use with pull-mode delivery.

5207 **R10.6-1:** A conformant service may implement the `SubscriptionEnd` message.

5208 **R10.6-2:** A conformant service shall not implement the `SubscriptionEnd` message when event  
5209 delivery is done using pull mode as defined in 10.2.9.4.



5210 **R10.6-3:** If SubscriptionEnd is supported, the message shall contain any reference parameters  
5211 specified by the subscriber in the EndTo address in the original subscription.

5212 **R10.6-4:** This rule intentionally left blank.

5213 If the service delivers events over the same connection as the Subscribe operation, the client typically  
5214 knows that a subscription has been terminated because the connection itself closes or terminates.

5215 When the delivery connection is distinct from the subscribe connection, a SubscriptionEnd message  
5216 is highly recommended; otherwise, the client has no immediate way of knowing that a subscription is  
5217 no longer active.

## 5218 **10.7 Acknowledgement of Delivery**

5219 To ensure that delivery is acknowledged at the application level, the original subscriber can request  
5220 that the event sink physically acknowledge event deliveries, rather than relying entirely on transport-  
5221 level guarantees.

5222 In other words, the transport might have accepted delivery of the events but not forwarded them to  
5223 the actual event sink process, and the service would move on to the next set of events. System  
5224 failures might result in dropped events. Therefore, a mechanism is needed in which a message-level  
5225 acknowledgement can occur. This allows acknowledgement to be pushed up to the application level,  
5226 increasing the reliability of event deliveries.

5227 The client selects acknowledged delivery by selecting a delivery mode in which each event has a  
5228 response. In this specification, the two acknowledged delivery modes are

- 5229 • <http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck>
- 5230 • <http://schemas.dmtf.org/wbem/wsman/1/wsman/Events>

5231 **R10.7-1:** A conformant service may support the PushWithAck or Events delivery mode.  
5232 However, if either of these delivery modes is requested, to maintain an ordered queue of events,  
5233 the service shall wait for the acknowledgement from the client before delivering the next event or  
5234 events that match the subscription.

5235 **R10.7-2:** If an acknowledged delivery mode is selected for the subscription, the service shall  
5236 include the following SOAP headers in each event delivery:

```
5237 (1) <s:Header>
5238 (2) <wsa:ReplyTo> where to send the acknowledgement </wsa:ReplyTo>
5239 (3) <wsman:AckRequested/>
5240 (4) ...
5241 (5) </s:Header>
```

5242 The following definitions provide additional, normative constraints on the preceding outline:

5243 **wsa:ReplyTo**

5244 address that shall always be present in the event delivery as a consequence of the presence of  
5245 **wsman:AckRequested**

5246 The client extracts this address and sends the acknowledgement to the specified EPR as  
5247 required by Addressing.

5248 **wsman:AckRequested**

5249 no content; requires that the subscriber acknowledge all deliveries as described later in this  
5250 clause

5251 The client then replies to the delivery with an acknowledgement or a fault.

5252 **R10.7-3:** A service may request receipt acknowledgement by using the wsman:AckRequested  
 5253 block and subsequently expect an `http://schemas.dmtf.org/wbem/wsman/1/wsman/Ack` message.  
 5254 If this message is not received as a reply, the service may terminate the subscription.

5255 The acknowledgement message format returned by the event sink (receiver) to the event source is  
 5256 identical for all delivery modes. As shown in the following outline, it contains a unique `wsa:Action`, and  
 5257 the `wsa:RelatesTo` field is set to the MessageID of the event delivery to which it applies:

```

5258 (1) <s:Envelope ...>
5259 (2) <s:Header>
5260 (3) ...
5261 (4) <wsa:To> endpoint reference from the event ReplyTo field </wsa:To>
5262 (5) <wsa:Action> http://schemas.dmtf.org/wbem/wsman/1/wsman/Ack
5263 (6) </wsa:Action>
5264 (7) <wsa:RelatesTo> message ID of original event delivery
5265 (8) </wsa:RelatesTo>
5266 (9) ...
5267 (10) </s:Header>
5268 (11) <s:Body/>
5269 (12) </s:Envelope>

```

5270 The following definitions provide additional, normative constraints on the preceding outline:

5271 `s:Envelope/s:Header/wsa:Action`

5272 URI that shall be defined as

5273 `http://schemas.dmtf.org/wbem/wsman/1/wsman/Ack`

5274 `s:Envelope/s:Header/wsa:RelatesTo`

5275 element that shall contain the `wsa:MessageID` of the event delivery to which it refers

5276 `wsa:RelatesTo` is the critical item that ensures that the correct delivery is being acknowledged,  
 5277 and thus it shall not be omitted.

5278 `s:Envelope/s:Header/wsa:To`

5279 EPR address extracted from the `ReplyTo` field in the event delivery

5280 All reference parameters shall be extracted and added to the SOAP header as well.

5281 In spite of the request to acknowledge, the event sink can refuse delivery with a fault or fail to  
 5282 respond with the acknowledgement. In this case, the event source can terminate the subscription and  
 5283 send any applicable `SubscriptionEnd` messages.

5284 If the event sink does not support acknowledgement, it can respond with a  
 5285 `wsman:UnsupportedFeature` fault with the following detail code:

5286 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Ack`

5287 However, this action is just as difficult as acknowledging the delivery, so most clients can scan for the  
 5288 `wsman:AckRequested` field and be prepared to acknowledge delivery or fault it.

## 5289 **10.8 Refusal of Delivery**

5290 With all acknowledged delivery modes as described in 10.7, an event sink can refuse to take delivery  
 5291 of events, either for security reasons or a policy change. It then responds with a fault rather than an  
 5292 acknowledgement.

5293 In this case, the event source needs to be prepared to end the subscription even though an  
 5294 `Unsubscribe` message is not issued by the subscriber.

5295 **R10.8-1:** During event delivery, if the receiver faults the delivery with a wsman:DeliveryRefused  
 5296 fault, the service shall immediately cancel the subscription and may also issue a SubscriptionEnd  
 5297 message to the EndTo endpoint in the original subscription, if supported.

5298 Thus, the receiver can issue the fault as a way to cancel the subscription when it does not have the  
 5299 SubscriptionManager information.

## 5300 10.9 Dropped Events

5301 Events that cannot be delivered are not to be silently dropped from the event stream, or the  
 5302 subscriber gets a false picture of the event history. WS-Management defines three behaviors for  
 5303 events that cannot be delivered with push modes or that are too large to fit within the delivery  
 5304 constraints requested by the subscriber:

- 5305 • Terminate the subscription.
- 5306 • Silently skip such events.
- 5307 • Send a special event in place of the dropped events.

5308 These options are discussed in 10.2.9.2 and 10.2.9.3.

5309 During delivery, the service might have to drop events for the following reasons:

- 5310 • The events exceed the maximum size requested by the subscriber.
- 5311 • The client cannot keep up with the event flow, and there is a backlog.
- 5312 • The service might have been reconfigured or restarted and the events permanently lost.

5313 In these cases, a service can inform the client that events have been dropped.

5314 **R10.9-1:** If a service drops events, it should issue an  
 5315 <http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents> event, which indicates this drop  
 5316 to the client. Any reference parameters specified in the NotifyTo address in the subscription shall  
 5317 also be copied into this message. This event is normal and implicitly considered part of any  
 5318 subscription.

5319 **R10.9-2:** If an <http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents> event is issued, it  
 5320 shall take the ordinal position of the original dropped event in the delivery stream. The  
 5321 DroppedEvents event is considered the same as any other event with regard to its location and  
 5322 other behavior (bookmarks, acknowledged delivery, location in batch, and so on). It simply takes  
 5323 the place of the event that was dropped.

5324 EXAMPLE:

```

5325 (1) <s:Envelope ...>
5326 (2) <s:Header>
5327 (3) ...subscriber endpoint-reference...
5328 (4)
5329 (5) <wsa:Action>
5330 (6) http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents
5331 (7) </wsa:Action>
5332 (8) </s:Header>
5333 (9) <s:Body>
5334 (10) <wsman:DroppedEvents Action="wsa:Action URI of dropped event">
5335 (11) xs:int
5336 (12) </wsman:DroppedEvents>
5337 (13) ...
5338 (14) </s:Body>

```

5339 (15) </s:Envelope>

5340 The following definitions provide additional, normative constraints on the preceding outline:

5341 s:Envelope/s:Header/wsa:Action

5342 URI that shall be defined as

5343 `http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents`

5344 s:Body/wsman:DroppedEvents/@Action

5345 the Action URI of the event that was dropped

5346 s:Body/wsman:DroppedEvents

5347 a positive integer that represents the total number of dropped events since the subscription was  
5348 created

5349 Renew has no effect on the running total of dropped events. Dropped events are like any other  
5350 events and can require acknowledgement, affect the bookmark location, and so on.

5351 EXAMPLE: Following is an example of how a dropped event would appear in the middle of a batched  
5352 event delivery:

```
5353 (1) <wsman:Events>
5354 (2) <wsman:Event Action="https://foo.com/someEvent">
5355 (3) ...event body
5356 (4) </wsman:Event>
5357 (5) <wsman:Event
5358 (6) Action="http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents">
5359 (7) <wsman:DroppedEvents Action="https://foo.com/someEvent">
5360 (8) 1
5361 (9) </wsman:DroppedEvents>
5362 (10) </wsman:Event>
5363 (11) <wsman:Event Action="https://foo.com/someEvent">
5364 (12) ...event body...
5365 (13) </wsman:Event>
5366 (14) </wsman:Events>
```

5367 **R10.9-3:** If a service cannot deliver an event and does not support the  
5368 `http://schemas.dmtf.org/wbem/wsman/1/wsman/DroppedEvents` event, it should terminate the  
5369 subscription rather than silently skipping events.

5370 Because this requirement cannot be enforced, and some dropped events are irrelevant when  
5371 replaced by a subsequent event (running totals, for example), it is not a firm requirement that dropped  
5372 events are signaled or that they result in a termination of the subscription.

## 5373 10.10 Access Control

5374 It is important for event sources to properly authorize requests. This is especially true for Subscribe  
5375 requests, because otherwise the ability to subscribe on behalf of a third-party event sink could be  
5376 used to create a distributed denial-of-service attack.

5377 Some possible schemes for validating Subscribe requests include:

- 5378 • Send a message to the event sink that describes the requested subscription, and then wait  
5379 for a confirmation message to be returned by the event sink, before the event source  
5380 accepts the subscription request. While this provides strong assurance that the event sink  
5381 actually desires the requested subscription, it does not work for event sinks that are not  
5382 capable of sending a confirmation, and requires additional logic on the event sink.

- 5383       • Require user authentication on the Subscribe request, and allow only authorized users to  
5384       Subscribe.

5385       Other mechanisms are also possible. Be aware that event sources that are not reachable from the  
5386       Internet have less need to control Subscribe requests.

## 5387       10.11 Implementation Considerations

5388       Implementations should generate expirations in Subscribe and Renew request and response  
5389       messages that are significantly larger than expected network latency.

5390       Event sinks should be prepared to receive notifications after sending a Subscribe request but before  
5391       receiving a Subscribe response message. Event sinks should also be prepared to receive  
5392       notifications after receiving an Unsubscribe response message.

## 5393       10.12 Advertisement of Notifications

5394       An Event Source can choose to advertise the Notification messages that it might send by including a  
5395       well-defined portType, called "EventSink", in its WSDL. Subscribers can examine this portType to  
5396       determine which messages they might need to support. Each Notification appears as an independent  
5397       operation within the portType, as shown in the following example:

5398       EXAMPLE:

```
5399 (1) <wsdl:portType name="EventSink">
5400 (2) <wsdl:operation name="WeatherReport">
5401 (3) <wsdl:input message="wr:ThunderStormMessage"
5402 (4) wsa:Action="urn:weatherReport:ThunderStorm"
5403 (5) wsam:Action="urn:weatherReport:ThunderStorm" />
5404 (6) <wsdl:input message="wr:TyphoonMessage"
5405 (7) wsa:Action="urn:weatherReport:Typhoon"
5406 (8) wsam:Action="urn:weatherReport:Typhoon" />
5407 (9) </wsdl:operation>
5408 (10) </wsdl:portType>
```

5409       In the preceding example this Event Source can send two types of Notifications (a ThunderStorm and a Typhoon  
5410       message).

5411       Unless otherwise noted, Event Sinks should assume that the Notifications will be sent using SOAP1.2  
5412       and will use document-literal encoding.

## 5413       11 Metadata and Discovery

5414       The WS-Management protocol is compatible with many techniques for discovery of resources  
5415       available through a service.

5416       In addition, this specification defines a simple request-response operation to facilitate the process of  
5417       establishing communications with a WS-Management service implementation in a variety of network  
5418       environments without prior knowledge of the protocol version or versions supported by the  
5419       implementation. This operation is used to discover the presence of a service that is compatible with  
5420       WS-Management, assuming that a transport address over which the message can be delivered is  
5421       known. Typically, a simple HTTP address would be used.

5422       To ensure forward compatibility, the message content of this operation is defined in an XML  
5423       namespace that is separate from the core protocol namespace and that will not change as the  
5424       protocol evolves. Further, this operation does not depend on any SOAP envelope header or body  
5425       content other than the types explicitly defined for this operation. In this way, WS-Management clients  
5426       are assured of the ability to use this operation against all implementations and versions to confirm the

5427 presence of WS-Management services without knowing the supported protocol versions or features in  
5428 advance.

5429 The request message is defined as follows:

```
5430 (1) <s:Envelope
5431 (2) xmlns:s="http://www.w3.org/2003/05/soap-envelope"
5432 (3) xmlns:wsmid="http://schemas.dmtf.org/wbem/wsman/identity/1/
5433 (4) wsmidentity.xsd"
5434 (4) <s:Header>
5435 (5) ...
5436 (6) </s:Header>
5437 (7) <s:Body>
5438 (8) <wsmid:Identify>
5439 (9) ...
5440 (10) </wsmid:Identify>
5441 (11) </s:Body>
5442 (12) </s:Envelope>
```

5443 The following definitions provide additional, normative constraints on the preceding outline:

5444 `wsmid:Identify`

5445 the body of the Identify request operation, which may contain additional vendor-specific  
5446 extension content, but is otherwise empty

5447 The presence of this body element constitutes the request.

5448 Notice the absence of any Addressing namespace, WS-Management namespace, or other version-  
5449 specific concepts. This message is compatible only with the [basic SOAP specification](#), and the  
5450 presence of the `wsmid:Identify` block in the `s:Body` is the embodiment of the request operation.

5451 The response message is defined as follows:

```
5452 (13) <s:Envelope
5453 (14) xmlns:s="http://www.w3.org/2003/05/soap-envelope"
5454 (15) xmlns:wsmid="http://schemas.dmtf.org/wbem/wsman/identity/1/
5455 (16) wsmidentity.xsd">
5456 (16) <s:Header>
5457 (17) ...
5458 (18) </s:Header>
5459 (19) <s:Body>
5460 (20) <wsmid:IdentifyResponse>
5461 (21) <wsmid:ProtocolVersion> xs:anyURI </wsmid:ProtocolVersion> +
5462 (22) <wsmid:ProductVendor> xs:string </wsmid:ProductVendor> ?
5463 (23) <wsmid:ProductVersion> xs:string </wsmid:ProductVersion> ?
5464 (24) <wsmid:InitiativeSupport>
5465 (25) <wsmid:InitiativeName> xs:string </wsmid:InitiativeName> ?
5466 (26) <wsmid:InitiativeVersion> xs:string </wsmid:InitiativeVersion> ?
5467 (27) </wsmid:InitiativeSupport> ?
5468 (28) <wsmid:SecurityProfiles>
5469 (29) <wsmid:SecurityProfileName> xs:anyURI
5470 (30) </wsmid:SecurityProfileName> *
5471 (30) </wsmid:SecurityProfiles> ?
5472 (31) <wsmid:AddressingVersionURI> xs:anyURI
5473 (32) </wsmid:AddressingVersionURI> *
5474 (32) ...
5475 (33) </wsmid:IdentifyResponse>
5476 (34) </s:Body>
5477 (35) </s:Envelope>
```

- 5478 The following definitions provide additional, normative constraints on the preceding outline:
- 5479 `wsmid:IdentifyResponse`
- 5480 the body of the response, which packages metadata about the WS-Management implementation
- 5481 `wsmid:IdentifyResponse/wsmid:ProtocolVersion`
- 5482 a required element or elements, each of which is a URI whose value shall be equal to the core
- 5483 XML namespace that identifies a supported version of the WS-Management specification
- 5484 One element shall be provided for each supported version of the protocol. Services should also
- 5485 include the XML namespace URI for supported dependent specifications such as Addressing.
- 5486 For example, if a future version of WS-Management supports multiple versions of Addressing,
- 5487 the `IdentifyResponse` can indicate which of the versions are supported.
- 5488 `wsmid:IdentifyResponse/wsmid:ProductVendor`
- 5489 an optional element that identifies the vendor of the WS-Management service implementation by
- 5490 using a widely recognized name or token, such as the official corporate name of the vendor or its
- 5491 stock symbol
- 5492 Alternatively, a DNS name, e-mail address, or Web URL may be used.
- 5493 `wsmid:IdentifyResponse/wsmid:ProductVersion`
- 5494 an optional version string for the WS-Management implementation
- 5495 This specification places no constraints on the format or content of this element.
- 5496 `wsmid:IdentifyResponse/wsmid:InitiativeSupport`
- 5497 an optional element that identifies an initiative supported by the WS-Management
- 5498 implementation.
- 5499 `wsmid:IdentifyResponse/wsmid:InitiativeSupport/wsmid:InitiativeName`
- 5500 an element that identifies the name of an initiative supported by the WS-Management
- 5501 implementation.
- 5502 `wsmid:IdentifyResponse/wsmid:InitiativeSupport/wsmid:InitiativeVersion`
- 5503 an element that identifies the version of an initiative supported by the WS-Management
- 5504 implementation.
- 5505 In addition, vendor-specific content can follow the preceding standardized elements. After the vendor-
- 5506 specific content, the following elements can follow:
- 5507 `wsmid:IdentifyResponse/wsmid:SecurityProfiles`
- 5508 an optional element that identifies the set of security profiles supported by the WS-Management
- 5509 implementation.
- 5510 `wsmid:IdentifyResponse/wsmid:SecurityProfiles/wsmid:SecurityProfileName`
- 5511 an optional element which is a URI that identifies a security profile supported by the WS-
- 5512 Management implementation.
- 5513 `wsmid:IdentifyResponse/wsmid:AddressingVersionURI`
- 5514 an optional element which is a URI that identifies a version of Addressing supported by the WS-
- 5515 Management implementation.
- 5516 When a service supports this element, the value shall be the XML Schema namespace URI of
- 5517 the addressing version in use. XML Schema namespaces used in this specification are listed in
- 5518 ANNEX A. A service may support and advertise more than none version of addressing.
- 5519 **R11-1:** A WS-Management service should support the `wsmid:Identify` operation. A service
- 5520 implementation that supports the operation shall do so irrespective of the versions of
- 5521 WS-Management supported by that service. The operation shall be accessible at the same

- 5522 transport-level address at which the resource instances are made accessible.
- 5523 It is recommended that client applications not include any SOAP header content in the wsmid:Identify  
5524 operation delivered to the transport address against which the inquiry is being made. If SOAP header  
5525 elements are present, the s:mustUnderstand attribute on all such elements can be set to "false".  
5526 Doing otherwise reduces the likelihood of a successful, version-independent response from the  
5527 service.
- 5528 **R11-2:** A service that supports the wsmid:Identify operation shall not require the presence of any  
5529 SOAP header elements in order to dispatch execution of the request. If a service receives a  
5530 wsmid:Identify operation that contains unexpected or unsupported header content with the  
5531 s:mustUnderstand attribute set to "false", the service shall not fault the request and shall process  
5532 the body of the request as though the header elements were not present.
- 5533 **R11-3:** A service that is processing the wsmid:Identify request should not request the presence  
5534 of any Addressing header values, including the wsa:Action URI.
- 5535 The entire purpose of this mechanism is to be able to identify the presence of specific versions of  
5536 WS-Management (and the corresponding dependent protocols) in a version-independent manner.
- 5537 Because Addressing is not used, the address to which this message is delivered is defined entirely at  
5538 the transport level and not present in the SOAP content.
- 5539 If a client does not have any prior knowledge about a service including credentials, it is desirable to  
5540 allow a service to process an Identify message without requiring authentication.
- 5541 **R11-4:** A service that supports the wsmid:Identify operation may expose this operation without  
5542 requiring client or server authentication in order to process the message. In the absence of other  
5543 requirements, it is recommended that the network address be suffixed by the token sequence  
5544 */wsman-anon/identify*.
- 5545 Services that support unauthenticated wsmid:Identify requests might choose not to reveal descriptive  
5546 information about protocol, vendor, or other versioning information that could potentially represent or  
5547 contribute to a vulnerability. To accommodate this scenario, this specification defines a URI that  
5548 services can use in place of a valid WS-Management protocol version URI. This value can be  
5549 returned as a value for the wsmid:ProtocolVersion element of the wsmid:IdentifyResponse message.
- 5550 **R11-5:** A service supporting an unauthenticated wsmid:Identify message may respond using the  
5551 following URI for the value of the wsmid:ProtocolVersion element:  
5552 <http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity/NoAnonymousDisclosure>
- 5553 **R11-6:** A service that provides unauthenticated access to the wsmid:Identify operation but does  
5554 not respond to such requests with the WS-Management protocol versions that are supported by  
5555 the service shall support authenticated access to the wsmid:Identify operation. Such services  
5556 shall respond to authenticated requests with the WS-Management protocol version identifiers for  
5557 each version of the WS-Management protocol supported by the service.

## 5558 12 Security

### 5559 12.1 General

- 5560 In general, management operations and responses need to be protected against attacks such as  
5561 snooping, interception, replay, and modification during transmission. Authenticating the user who has  
5562 sent a request is also generally necessary so that access control rules can be applied to determine  
5563 whether to process a request.



5564 This specification establishes the minimum interoperation standards and predefined profiles using  
5565 transport-level security.

5566 This approach provides the best balance between simple implementations (HTTP and HTTPS stacks  
5567 are readily available, even for hardware) and the security mechanisms that sit in front of any SOAP  
5568 message processing, limiting the attack surface.

5569 More sophisticated transport and SOAP-level profiles, published separately from this specification,  
5570 may be defined and used.

5571 Implementations that expect to interoperate can adopt one or more of the transport and security  
5572 models defined in this clause and are free to define any additional profiles under different URI-based  
5573 designators.

## 5574 **12.2 Security Profiles**

5575 For this specification, a profile is any arbitrary mix of transport or SOAP behavior that describes a  
5576 common security need. In some cases, the profile is defined for documentation and metadata  
5577 purposes, but might not be part of the actual message exchange. Rather, it *describes* the message  
5578 exchange involved.

5579 Metadata retrieval can be employed to discover which profiles the service supports, and that is  
5580 beyond the scope of this particular specification.

5581 For all predefined profiles, the transport is responsible for all message integrity, protection,  
5582 authentication, and security.

5583 This specification makes no assumptions about the security requirements of the applications that use  
5584 WS-Eventing. However, once those requirements have been satisfied within a given operational  
5585 context, the addition of WS-Eventing to this operational context cannot undermine the fulfillment of  
5586 those requirements; the use of WS-Eventing SHOULD NOT create additional attack vectors within an  
5587 otherwise secure system.

5588 The authentication profiles do not appear in the SOAP traffic, with the exception of the Subscribe  
5589 message when using any delivery mode that causes a new connection to be created from the event  
5590 source to the event sink (push and batched modes, for example). When a subscription is created, the  
5591 authentication technique for event-delivery needs to be specified by the subscriber, because the  
5592 event sink has to authenticate the event source (acting as publisher) at event delivery-time.

5593 In this specification, security profiles are identified by a URI. As profiles are defined, they can be  
5594 assigned a URI and published. WS-Management defines a set of standardized security profiles for  
5595 the common transports HTTP and HTTPS as described in C.3.1.

## 5596 **12.3 Security Considerations for Event Subscriptions**

5597 When specifying the NotifyTo address in subscriptions, it is often important to hint to the service  
5598 about which authentication model to use when delivering the event.

5599 If no hints are present, the service can simply infer from the wsa:To address what needs to be done.  
5600 However, if the service can support multiple modes and has a certificate or password store, it might  
5601 not know which authentication model to choose or which credentials to use without being told in the  
5602 subscription.

5603 WS-Management provides a default mechanism to communicate the desired authentication mode  
5604 and credentials. However, more sophisticated mechanisms are beyond the scope of this version of  
5605 WS-Management. For example, the event sink service could export metadata that describes the  
5606 available options, allowing the publisher to negotiate an appropriate option. Extension profiles can  
5607 define other mechanisms enabled through a SOAP header with mustUnderstand="true".

5608 WS-Management defines an additional field in the Delivery block that can communicate  
5609 authentication information, as shown in the following outline:

```
5610 (1) <s:Body>
5611 (2) <wsme:Subscribe>
5612 (3) <wsme:Delivery>
5613 (4) <wsme:NotifyTo> Delivery EPR </wsme:NotifyTo>
5614 (5) <wsman:Auth Profile="authentication-profile-URI"/>
5615 (6) </wsme:Delivery>
5616 (7) </wsme:Subscribe>
5617 (8) </s:Body>
```

5618 The following definitions provide additional, normative constraints on the preceding outline:

5619 wsman:Auth

5620 block that contains authentication information to be used by the service (acting as publisher)  
5621 when authenticating to the event sink at event delivery time

5622 wsman:Auth/@Profile

5623 a URI that indicates which security profile to use when making the connection to deliver events

5624 If the wsman:Auth block is not present, by default the service infers what to do by using the NotifyTo  
5625 address and any preconfigured policy or settings it has available. If the wsman:Auth block is present  
5626 and no security-related tokens are communicated, the service needs to know which credentials to use  
5627 by its own internal configuration.

5628 If the service is already configured to use a specific certificate when delivering events, the subscriber  
5629 can request standard mutual authentication, as shown in the following outline:

```
5630 (1) <s:Body>
5631 (2) <wsme:Subscribe>
5632 (3) <wsme:Delivery>
5633 (4) <wsme:NotifyTo> HTTPS address </wsme:NotifyTo>
5634 (5) <wsman:Auth
5635 (6) Profile="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/
5636 (7) mutual"/>
5637 (7) </wsme:Delivery>
5638 (8) </wsme:Subscribe>
5639 (9) </s:Body>
```

5640 If the service knows how to retrieve a proper user name and password for event delivery, simple  
5641 HTTP Basic or Digest authentication can be used, as shown in the following outline:

```
5642 (1) <s:Body>
5643 (2) <wsme:Subscribe>
5644 (3) <wsme:Delivery>
5645 (4) <wsme:NotifyTo> HTTP address </wsme:NotifyTo>
5646 (5) <wsman:Auth
5647 (6) Profile="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/
5648 (7) digest"/>
5649 (7) </wsme:Delivery>
5650 (8) </wsme:Subscribe>
5651 (9) </s:Body>
```

5652 Services are not required to support any specific profile. The rest of this clause defines special-case  
5653 profiles for event delivery in which the service needs additional information to select the proper  
5654 credentials to use when delivering events.

## 5655 12.4 Including Credentials with a Subscription

5656 This clause intentionally left blank.

## 5657 12.5 Correlating Events with a Subscription

5658 In many cases, the subscriber will want to ensure that the event delivery corresponds to a valid  
5659 subscription issued by an authorized party. In this case, it is recommended that reference parameters  
5660 be introduced into the NotifyTo definition.

5661 EXAMPLE: At subscription time, a UUID could be supplied as a correlation token:

```
5662 (1) <s:Body>
5663 (2) <wsme:Subscribe>
5664 (3) <wsme:Delivery>
5665 (4) <wsme:NotifyTo>
5666 (5) <wsa:Address address <wsa:Address>
5667 (6) <wsa:ReferenceParameters>
5668 (7) <MyNamespace:uuid>
5669 (8) uuid:b0f685ec-e5c9-41b5-b91c-7f580419093e
5670 (9) </MyNamespace:uuid>
5671 (10) </wsa:ReferenceParameters>
5672 (11) </wsme:NotifyTo>
5673 (12) ...
5674 (13) </wsme:Delivery>
5675 (14) ...
5676 (15) </wsme:Subscribe>
5677 (16) </s:Body>
```

5678 This definition requires that the service include the MyNamespace:uuid value as a SOAP header with  
5679 each event delivery (see 5.1). The service can use this value to correlate the event with any  
5680 subscription that it issued and to validate its origin.

5681 This is not a transport-level or SOAP-level authentication mechanism as such, but it does help to  
5682 maintain and synchronize valid lists of subscriptions and to determine whether the event delivery is  
5683 authorized, even though the connection itself could have been authenticated.

5684 This mechanism still can require the presence of the wsman:Auth block to specify which security  
5685 mechanism to use to actually authenticate the connection at event-time.

5686 Each new subscription can receive at least one unique reference parameter that is never reused,  
5687 such as the illustrated UUID, for this mechanism to be of value.

5688 Other reference parameters can be present to help route and correlate the event delivery as required  
5689 by the subscriber.

## 5690 12.6 Transport-Level Authentication Failure

5691 Because transports typically go through their own authentication mechanisms prior to any SOAP  
5692 traffic occurring, the first attempt to connect might result in a transport-level authentication failure. In  
5693 such cases, SOAP faults will not occur, and the means of communicating the denial to the client is  
5694 implementation- and transport-specific.

## 5695 12.7 Security Implications of Third-Party Subscriptions

5696 Without proper authentication and authorization, WS-Management implementations can be  
5697 vulnerable to distributed denial-of-service attacks through third-party subscriptions to events. This  
5698 vulnerability is discussed in 10.10.

## 5699 13 Transports and Message Encoding

5700 This clause describes encoding rules that apply to all transports.

### 5701 13.1 SOAP

5702 WS-Management qualifies the use of SOAP as indicated in this clause.

5703 **R13.1-1:** A service shall at least receive and send [SOAP 1.2](#) SOAP Envelopes.

5704 **R13.1-2:** A service may reject a SOAP Envelope with more than 32,767 octets.

5705 **R13.1-3:** A service should not send a SOAP Envelope with more than 32,767 octets unless the  
5706 client has specified a wsman:MaxEnvelopeSize header that overrides this limit.

5707 Large SOAP Envelopes are expected to be serialized using attachments.

5708 **R13.1-4:** Any Request Message may be encoded using either Unicode 3.0 (UTF-16) or UTF-8  
5709 encoding. A service shall accept the UTF-8 encoding type for all operations and should accept  
5710 UTF-16 as well.

5711 **R13.1-5:** A service shall emit Responses using the same encoding as the original request. If the  
5712 service does not support the requested encoding or cannot determine the encoding, it should use  
5713 UTF-8 encoding to return a wsman:EncodingLimit fault with the following detail code:

5714 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/CharacterSet`

5715 **R13.1-6:** For UTF-8 encodings, the service may fail to process any message that begins with the  
5716 UTF-8 BOM (0xEF 0xBB 0xBF), and shall send UTF-8 responses without the BOM.

5717 The presence of BOM in 8-bit character encodings reduces interoperability. Where extended  
5718 characters are a requirement, UTF-16 can be used.

5719 **R13.1-7:** If UTF-16 is the encoding, the service shall support either byte-order mark (BOM)  
5720 U+FEFF (big-endian) or U+FFFE (little-endian) as defined in the [Unicode 3.0](#) specification as the  
5721 first character in the message (see the [Unicode BOM FAQ](#)).

5722 **R13.1-8:** If a request includes contradictory encoding information in the BOM and HTTP charset  
5723 header or if the information does not fully specify the encoding, the service shall fault with an  
5724 HTTP status of "bad request message" (400).

5725 Repeated headers with the same QName but different values that imply contradictory behavior are  
5726 considered a defect originating on the client side of the conversation. Returning a fault helps identify  
5727 faulty clients. However, an implementation might be resource-constrained and unable to detect  
5728 duplicate headers, so the repeated headers can be ignored. Repeated headers with the same  
5729 QName that contains informational or non-contradictory instructions are possible, but none are  
5730 defined by this specification or its dependencies.

5731 **R13.1-9:** If a request contains multiple SOAP headers with the same QName from  
5732 WS-Management, Addressing, or clause 10 of this specification, the service should not process  
5733 them and should issue a ws:InvalidMessageInformationHeaders fault if they are detected. (No  
5734 SOAP headers are defined in clause 7 "Resource Access" or clause 8 "Enumeration of  
5735 Datasets".)

5736 **R13.1-10:** By default, a compliant service should not fault requests with leading and trailing  
5737 whitespace in XML element values and should trim such whitespace by default as if the  
5738 whitespace had not occurred. Services should not emit messages containing leading or trailing

5739 whitespace within element values unless the whitespace values are properly part of the value. If  
5740 the service cannot accept whitespace usage within a message because the XML schema  
5741 establishes other whitespace usage, the service should emit a wsman:EncodingLimit fault with  
5742 the following detail code:

5743 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Whitespace`

5744 Clients can send messages with leading or trailing whitespace in the values, and services are  
5745 permitted to eliminate unneeded "cosmetic" whitespace on both sides of the element value without  
5746 faulting. (See [XML Schema Part 2: Datatypes](#).)

5747 **R13.1-11:** Services should not fault messages that contain XML comments, because this is part  
5748 of the XML standard. Services may emit messages that contain comments that relate to the origin  
5749 and processing of the message or add comments for debugging purposes.

## 5750 13.2 Lack of Response

5751 If an operation succeeds but a response cannot be computed or actually delivered because of run-  
5752 time difficulties or transport problems, no response is sent and the connection is terminated.

5753 This behavior is preferable to attempting a complex model for sending responses in a delayed  
5754 fashion. Implementations can generally keep a log of all requests and their results, and allow the  
5755 client to reconnect later to enumerate the operation log (using Enumerate) if it failed to get a  
5756 response. The format and behavior of such a log is beyond the scope of this specification. In any  
5757 case, the client needs to be coded to take into account a lack of response; all abnormal message  
5758 conditions can safely revert to this scenario.

5759 **R13.2-1:** If correct responses or faults cannot be computed or generated due to internal service  
5760 failure, a response should not be sent.

5761 Regardless, the client has to deal with cases of no response, so the service can simply force the  
5762 client into that mode rather than send a response or fault that is not defined in this specification.

## 5763 13.3 Replay of Messages

5764 This section intentionally left blank.

5765 **R13.3-1:** This rule intentionally left blank.

## 5766 13.4 Encoding Limits

5767 Most of the following limits are in characters. However, the maximum overall SOAP envelope size is  
5768 defined in octets. Implementations are free to exceed these limits. A service is considered conformant  
5769 if it observes these limits. Any limit violation results in a wsman:EncodingLimit fault.

5770 **R13.4-1:** A service may fail to process any URI with more than 2048 characters and should  
5771 return a wsman:EncodingLimit fault with the following detail code:

5772 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/URILimitExceeded`

5773 **R13.4-2:** A service should not generate a URI with more than 2048 characters.

5774 **R13.4-3:** A service may fail to process an Option Name of more than 2048 characters.

5775 **R13.4-4:** A service may fail to process an Option value of more than 4096 characters.

5776 **R13.4-5:** A service may fault any operation that would require a single reply exceeding 32,767  
5777 octets.

5778 **R13.4-6:** A service may always emit faults that are 4096 octets or less in length, regardless of  
5779 any requests by the client to limit the response size. Clients need to be prepared for this minimum  
5780 in case of an error.

5781 **R13.4-7:** When the default addressing model is in use, a service may fail to process a Selector  
5782 Name of more than 2048 characters.

5783 **R13.4-8:** A service may have a maximum number of selectors that it can process. If the request  
5784 contains more selectors than this limit, the service should return a wsman:EncodingLimit fault  
5785 with the following detail code:

5786 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/SelectorLimit`

5787 **R13.4-9:** A service may have a maximum number of options that it can process. If the request  
5788 contains more options than this limit, the service should return a wsman:EncodingLimit fault with  
5789 the following detail code:

5790 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OptionLimit`

### 5791 **13.5 Binary Attachments**

5792 SOAP Message Transmission Optimization Mechanism (MTOM) is used to support binary  
5793 attachments to WS-Management. If a service supports attachments, the following rules apply:

5794 **R13.5-1:** A conformant service may optionally support binary attachments to any operation using  
5795 the [SOAP MTOM](#) proposal.

5796 **R13.5-2:** If a service supports attachments, the service shall support the Abstract Transmission  
5797 Optimization Feature.

5798 **R13.5-3:** If a service supports attachments, the service shall support the Optimized MIME  
5799 Multipart Serialization Feature.

5800 Other attachment types are not prohibited. Specific transports can impose additional encoding rules.

### 5801 **13.6 Case-Sensitivity**

5802 While XML and SOAP are intrinsically case-sensitive with regard to schematic elements,  
5803 WS-Management can be used with many underlying systems that are not intrinsically case-sensitive.  
5804 This support primarily applies to values, but can also apply to schemas that are automatically and  
5805 dynamically generated from other sources.

5806 A service can observe any case usage required by the underlying execution environment.

5807 The only requirement is that messages are able to pass validation tests against any schema  
5808 definitions. At any time, a validation engine could be interposed between the client and server in the  
5809 form of a proxy, so schematically valid messages are a practical requirement.

5810 Otherwise, this specification makes no requirements as to case usage. A service is free to interpret  
5811 values in a case-sensitive or case-insensitive manner.

5812 It is recommended that case usage not be altered in transit by any part of the WS-Management  
5813 processing chain. The case usage established by the sender of the message is to be retained  
5814 throughout the lifetime of that message.

5815 **14 Faults**

5816 Many of the operations outlined in WS-Management can generate faults. This clause describes how  
5817 these faults should be formatted into SOAP messages.

5818 **14.1 Introduction**

5819 Faults are returned when the SOAP message is successfully delivered by the transport and  
5820 processed by the service, but the message cannot be processed properly. If the transport cannot  
5821 successfully deliver the message to the SOAP processor, a transport error occurs.

5822 **R14.1-1:** A service should support only [SOAP 1.2](#) (or later) faults.

5823 Generally, faults are not to be issued unless they are expected as part of a request-response pattern.  
5824 For example, it would not be valid for a client to issue a Get message, receive the GetResponse  
5825 message, and then *fault* that response.

5826 **14.2 Fault Encoding**

5827 This clause discusses XML fault encoding.

5828 **R14.2-1:** A conformant service shall use the following fault encoding format and normative  
5829 constraints for faults in the WS-Management space or any of its dependent specifications:

```

5830 (1) <s:Envelope>
5831 (2) xmlns:s="http://www.w3.org/2003/05/soap-envelope"
5832 (3) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
5833 (4) <s:Header>
5834 (5) <wsa:Action>
5835 (6) http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
5836 (7) </wsa:Action>
5837 (8) <wsa:MessageID>
5838 (9) uuid:d9726315-bc91-430b-9ed8-ce5ffb858a87
5839 (10) </wsa:MessageID>
5840 (11) <wsa:RelatesTo>
5841 (12) uuid:d9726315-bc91-430b-9ed8-ce5ffb858a85
5842 (13) </wsa:RelatesTo>
5843 (14) </s:Header>
5844 (15)
5845 (16) <s:Body>
5846 (17) <s:Fault>
5847 (18) <s:Code>
5848 (19) <s:Value> [Code] </s:Value>
5849 (20) <s:Subcode>
5850 (21) <s:Value> [Subcode] </s:Value>
5851 (22) </s:Subcode>
5852 (23) </s:Code>
5853 (24) <s:Reason>
5854 (25) <s:Text xml:lang="en"> [Reason] </s:Text>
5855 (26) </s:Reason>
5856 (27) <s:Detail>
5857 (28) [Detail]
5858 (29) </s:Detail>
5859 (30) </s:Fault>
5860 (31) </s:Body>
5861 (32) </s:Envelope>

```

- 5862 The following definitions provide additional, normative constraints on the preceding outline:
- 5863 s:Envelope/s:Header/wsa:Action  
5864 a valid fault Action URI from the relevant specification that defined the fault
- 5865 s:Envelope/s:Header/wsa:MessageId  
5866 element that shall be present for the fault, like any non-fault message
- 5867 s:Envelope/s:Header/wsa:RelatesTo  
5868 element that shall, like any other reply, contain the MessageID of the original request that  
5869 caused the fault
- 5870 s:Body/s:Fault/s:Value  
5871 element that shall be either s:Sender or s:Receiver, as specified in 14.6 in the "Code" field
- 5872 s:Body/s:Fault/s:Subcode/s:Value  
5873 for WS-Management-related messages, shall be one of the subcode QNames defined in 14.6  
5874 If the service exposes custom methods or other messaging, this value may be another QName  
5875 not in the Master Faults described in 14.6.
- 5876 s:Body/s:Fault/s:Reason  
5877 optional element that should contain localized text that explains the fault in more detail  
5878 Typically, this text is extracted from the "Reason" field in the Master Fault tables (14.6).  
5879 However, the text may be adjusted to reflect a specific circumstance. This element may be  
5880 repeated for multiple languages. The xml:lang attribute shall be present on the s:Text element.
- 5881 s:Body/s:Fault/s:Detail  
5882 optional element that should reflect the recommended content from the Master Fault tables  
5883 (14.6)
- 5884 The preceding fault template is populated by examining entries from the Master Fault tables in 14.6,  
5885 which includes all relevant faults from WS-Management and its underlying specifications.
- 5886 s:Reason and s:Detail are always optional, but they are recommended. In addition, s:Reason/s:Text  
5887 contains an xml:lang attribute to indicate the language used in the descriptive text.
- 5888 **R14.2-2:** Fault wsa:Action URI values vary from fault to fault. The service shall issue a fault  
5889 using the correct URI, based on the specification that defined the fault. Faults defined in this  
5890 specification shall have the following URI value:
- 5891 `http://schemas.dmtf.org/wbem/wsman/1/wsman/fault`
- 5892 The Master Fault tables in 14.6 contain the relevant wsa:Action URIs. The URI values are directly  
5893 implied by the QName for the fault.

### 5894 14.3 NotUnderstood Faults

- 5895 There is a special case for faults relating to mustUnderstand attributes on SOAP headers. SOAP  
5896 specifications define the fault differently than the encoding in 14.2 (see 5.4.8 in [SOAP 1.2](#)). In  
5897 practice, the fault varies only in indicating the SOAP header that was not understood, the QName,  
5898 and the namespace (see line 5 in the following outline).

```
5899 (1) <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
5900 (2) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
5901 (3)
5902 (4) <s:Header>
5903 (5) <s:NotUnderstood qname="QName of header" xmlns:ns="XML namespace of
5904 header"/>
```



```

5905 (6) <wsa:Action>
5906 (7) http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
5907 (8) </wsa:Action>
5908 (9) <wsa:MessageID>
5909 (10) urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a87
5910 (11) </wsa:MessageID>
5911 (12) <wsa:RelatesTo>
5912 (13) urn:uuid:d9726315-bc91-430b-9ed8-ce5ffb858a85
5913 (14) </wsa:RelatesTo>
5914 (15) </s:Header>
5915 (16)
5916 (17) <s:Body>
5917 (18) <s:Fault>
5918 (19) <s:Code>
5919 (20) <s:Value>s:MustUnderstand</s:Value>
5920 (21) </s:Code>
5921 (22) <s:Reason>
5922 (23) <s:Text xml:lang="en-US">Header not understood</s:Text>
5923 (24) </s:Reason>
5924 (25) </s:Fault>
5925 (26) </s:Body>
5926 (27)
5927 (28) </s:Envelope>

```

5928 The preceding fault template can be used in all cases of failure to process mustUnderstand attributes.  
 5929 Lines 5–8 show the important content, indicating which header was not understood and including a  
 5930 generic wsa:Action that specifies that the current message is a fault.

5931 The wsa:RelatesTo element is included so that the client can correlate the fault with the original  
 5932 request. Over transports other than HTTP in which requests might be interlaced, this might be the  
 5933 only way to respond to the correct sender.

5934 If the original wsa:MessageID itself is faulty and the connection is request-response oriented, the  
 5935 service can attempt to send back a fault without the wsa:RelatesTo field, or can simply fail to  
 5936 respond, as discussed in 14.4.

#### 5937 **14.4 Degenerate Faults**

5938 In rare cases, the SOAP message might not contain enough information to properly generate a fault.  
 5939 For example, if the wsa:MessageID is garbled, the service will have difficulty returning a fault that  
 5940 references the original message. Some transports might not be able to reference the sender to return  
 5941 the fault.

5942 If the transport guarantees a simple request-response pattern, the service can send back a fault with  
 5943 no wsa:RelatesTo field. However, in some cases, there is no guarantee that the sender can be  
 5944 reached (for example, if the wsa:FaultTo contains an invalid address, so there is no way to deliver the  
 5945 fault).

5946 In all cases, the service can revert to the rules of 13.3, in which no response is sent. The service can  
 5947 attempt to log the requests in some way to help identify the defective client.

#### 5948 **14.5 Fault Extensibility**

5949 A service can include additional fault information beyond what is defined in this specification. The  
 5950 appropriate extension element is the s:Detail element, and the service-specific XML can appear at  
 5951 any location within this element, provided that it is properly mapped to an XML namespace that  
 5952 defines the schema for that content. WS-Management makes use of this extension technique for the  
 5953 wsman:FaultDetail URI values, as shown in the following outline:

```

5954 (1) <s:Detail>
5955 (2) <wsman:FaultDetail>... </wsman:FaultDetail>
5956 (3) <ExtensionData xmlns="vendor-specific-namespace">...</ExtensionData>
5957 (4) ...
5958 (5) </s:Detail>

```

5959 The extension data elements can appear before or after any WS-Management-specific extensions  
5960 mandated by this specification. More than one extension element is permitted.

## 5961 14.6 Master Faults

5962 This clause includes all faults from this specification and all underlying specifications. This list is the  
5963 normative fault list for WS-Management.

5964 **R14.6-1:** A service shall return faults from the following list when the operation that caused them  
5965 was a message in this specification for which faults are specified. A conformant service may  
5966 return other faults for messages that are not part of WS-Management.

5967 It is critical to client interoperability that the same fault be used in identical error cases. If each service  
5968 returns a distinct fault for "Not Found", for example, constructing interoperable clients would be  
5969 impossible. In Table 5 through Table 43, the source specification of a fault is based on its QName.

5970 **Table 5 – wsman:AccessDenied**

|               |                                                                                                                                                                                                                                             |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:AccessDenied                                                                                                                                                                                                                          |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                            |
| Code          | s:Sender                                                                                                                                                                                                                                    |
| Reason        | The sender was not authorized to access the resource.                                                                                                                                                                                       |
| Detail        | None                                                                                                                                                                                                                                        |
| Comments      | This fault is returned generically for all access denials that relate to authentication or authorization failures. This fault does not indicate locking or concurrency conflicts or other types of denials unrelated to security by itself. |
| Applicability | Any message                                                                                                                                                                                                                                 |
| Remedy        | The client acquires the correct credentials and retries the operation.                                                                                                                                                                      |

5971

Table 6 – wsa:ActionNotSupported

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsa:ActionNotSupported                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/addressing/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reason        | The action is not supported by the service.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsa:Action&gt; <i>Incorrect Action URI</i> &lt;/wsa:Action&gt; &lt;/s:Detail&gt; &lt;!-- The unsupported Action URI is returned, if possible --&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Comments      | <p>This fault means that the requested action is not supported by the implementation. As an example, read-only implementations (supporting only Get and Enumerate) return this fault for any operations other than these two.</p> <p>If the implementation never supports the action, the fault can be generated as shown in the "Detail" row of this table. However, if the implementation supports the action in a general sense, but it is not an appropriate match for the resource, an additional detail code can be added to the fault, as follows:</p> <pre>&lt;s:Detail&gt;   &lt;wsa:Action&gt; <i>The offending Action URI</i> &lt;/wsa:Action&gt;   &lt;wsman:FaultDetail&gt;     http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ActionMismatch   &lt;/wsman:FaultDetail&gt; &lt;/s:Detail&gt;</pre> <p>This situation can occur when the implementation supports Put, for example, but the client attempts to update a read-only resource.</p> |
| Applicability | All messages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Remedy        | The client consults metadata provided by the service to determine which operations are supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

5972

Table 7 – wsman:AlreadyExists

|               |                                                                                                    |
|---------------|----------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:AlreadyExists                                                                                |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                   |
| Code          | s:Sender                                                                                           |
| Reason        | The sender attempted to create a resource that already exists.                                     |
| Detail        | None                                                                                               |
| Comments      | This fault is returned in cases where the user attempted to create a resource that already exists. |
| Applicability | Create                                                                                             |
| Remedy        | The client uses Put or creates a resource with a different identity.                               |

5973

Table 8 – wsmen:CannotProcessFilter

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:CannotProcessFilter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reason        | The requested filter could not be processed.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsman:SupportedSelectorName&gt; Valid selector name for use in filter expression &lt;/wsman:SupportedSelectorName&gt; * &lt;/s:Detail&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Comments      | <p>This fault is returned for syntax errors or other semantic problems with the filter.</p> <p>For use with the SelectorFilter dialect (see ANNEX E), the service can include one or more SupportedSelectorName elements to provide a list of supported selector names in the event that the client has requested filtering on one or more unsupported selector names.</p> <p>If the filter is valid, but the service cannot execute the filter due to misconfiguration, lack of resources, or other service-related problems, more specific faults can be returned, such as wsman:QuotaLimit or wsman:InternalError.</p> |
| Applicability | Enumerate                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Remedy        | The client fixes the filter problem and tries again.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

5974

Table 9 – wsman:CannotProcessFilter

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:CannotProcessFilter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reason        | The requested filter could not be processed.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsman:SupportedSelectorName&gt; Valid selector name for use in filter expression &lt;/wsman:SupportedSelectorName&gt; * &lt;/s:Detail&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Comments      | <p>This fault is returned for syntax errors or other semantic problems with the filter such as exceeding the subset supported by the service.</p> <p>For use with the SelectorFilter dialect (see ANNEX E), the service can include one or more SupportedSelectorName elements to provide a list of supported selector names in the event that the client has requested filtering on one or more unsupported selector names.</p> <p>If the filter is valid, but the service cannot execute the filter due to misconfiguration, lack of resources, or other service-related problems, more specific faults can be returned, such as wsman:QuotaLimit, wsman:InternalError, or wsme:EventSourceUnableToProcess.</p> |
| Applicability | Subscribe, fragment-level resource access operations                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Remedy        | The client fixes the filter problem and tries again.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

5975

Table 10 – wsman:Concurrency

|               |                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:Concurrency                                                                                                                                                                                                                                                                                                                                                         |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                          |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                  |
| Reason        | The action could not be completed due to concurrency or locking problems.                                                                                                                                                                                                                                                                                                 |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                      |
| Comments      | This fault means that the requested action could not be carried out either due to internal concurrency or locking problems or because another user is accessing the resource. This fault can occur if a resource is being enumerated using Enumerate and another client attempts operations such as Delete, which would affect the result of the enumeration in progress. |
| Applicability | All messages                                                                                                                                                                                                                                                                                                                                                              |
| Remedy        | The client waits and tries again.                                                                                                                                                                                                                                                                                                                                         |

5976

Table 11 – wsme:DeliveryModeRequestedUnavailable

|               |                                                                                                                                                                                                                                                                                                                                                 |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:DeliveryModeRequestedUnavailable                                                                                                                                                                                                                                                                                                           |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                                                                                                                                                                                                                                            |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                        |
| Reason        | The requested delivery mode is not supported.                                                                                                                                                                                                                                                                                                   |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsme:SupportedDeliveryMode&gt;... &lt;/wsme:SupportedDeliveryMode&gt;   &lt;wsme:SupportedDeliveryMode&gt;...&lt;/wsme:SupportedDeliveryMode&gt;   ... &lt;/s:Detail&gt;</pre> <p>&lt;!-- This is a simple, optional list of one or more supported delivery mode URIs. It may be left empty. --&gt;</p>             |
| Comments      | This fault is returned for unsupported delivery modes for the specified resource. If the stack supports the delivery mode in general, but not for the specific resource, this fault is still returned. Other resources might support the delivery mode. The fault does not imply that the delivery mode is not supported by the implementation. |
| Applicability | Subscribe                                                                                                                                                                                                                                                                                                                                       |
| Remedy        | The client selects one of the supported delivery modes.                                                                                                                                                                                                                                                                                         |

5977

**Table 12 – wsman:DeliveryRefused**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:DeliveryRefused                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                             |
| Code          | s:Receiver                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reason        | The receiver refuses to accept delivery of events and requests that the subscription be canceled.                                                                                                                                                                                                                                                                                                                                                            |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Comments      | This fault is returned by event receivers to force a cancellation of a subscription.<br>This fault can happen when the client tried to Unsubscribe, but failed, or when the client lost knowledge of active subscriptions and does not want to keep receiving events that it no longer owns. This fault can help clean up spurious or leftover subscriptions when clients are reconfigured or reinstalled and their previous subscriptions are still active. |
| Applicability | Any event delivery message in any mode                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Remedy        | The service stops delivering events for the subscription and cancels the subscription, sending any applicable SubscriptionEnd messages.                                                                                                                                                                                                                                                                                                                      |

5978

**Table 13 – wsa:DestinationUnreachable**

|               |                                                                                                                                                                                                                                                                                                                            |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsa:DestinationUnreachable                                                                                                                                                                                                                                                                                                 |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/addressing/fault                                                                                                                                                                                                                                                                     |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                   |
| Reason        | No route can be determined to reach the destination role defined by the Addressing To header.                                                                                                                                                                                                                              |
| Detail        | <s:Detail><br><wsman:FaultDetail><br>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidResourceURI<br></wsman:FaultDetail> ?<br></s:Detail><br>When the default addressing model is in use, the wsman:FaultDetail field may contain http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidResourceURI. |
| Comments      | This fault is returned as the general "Not Found" case for a resource, in which the resource EPR cannot be mapped to the real-world resource.<br>This fault is not used merely to indicate that the resource is temporarily offline, which is indicated by wsa:EndpointUnavailable.                                        |
| Applicability | All request messages                                                                                                                                                                                                                                                                                                       |
| Remedy        | The client attempts to diagnose the version of the service, query any metadata, and perform other diagnostic operations to determine why the request cannot be routed.                                                                                                                                                     |

5979

Table 14 – wsman:EncodingLimit

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:EncodingLimit                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Reason        | An internal encoding limit was exceeded in a request or would be violated if the message were processed.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Detail        | <p>&lt;s:Detail&gt;<br/>         &lt;wsman:FaultDetail&gt;<br/>         Optional; one of the following enumeration values<br/>         &lt;/wsman:FaultDetail&gt;<br/>         ...any service-specific additional XML content...<br/>         &lt;/s:Detail&gt;</p> <p>Possible enumeration values in the &lt;wsman:FaultDetail&gt; element are as follows:</p> <p>Unsupported character set:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/CharacterSet">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/CharacterSet</a></p> <p>Unsupported MTOM or other encoding types:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EncodingType">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EncodingType</a></p> <p>Requested maximum was too large:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize</a></p> <p>Requested maximum envelope size was too small:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MinimumEnvelopeLimit">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MinimumEnvelopeLimit</a></p> <p>Too many options:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OptionLimit">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OptionLimit</a></p> <p>Used when the default addressing model is in use and indicates that too many selectors were used for the corresponding ResourceURI:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/SelectorLimit">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/SelectorLimit</a></p> <p>Service reached its own internal limit when computing response:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ServiceEnvelopeLimit">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ServiceEnvelopeLimit</a></p> <p>Operation succeeded and cannot be reversed, but result is too large to send:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnreportableSuccess">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnreportableSuccess</a></p> <p>Request contained a character outside of the range that is supported by the service:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnsupportedCharacter">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnsupportedCharacter</a></p> <p>URI was too long:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/URILimitExceeded">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/URILimitExceeded</a></p> <p>Client-side whitespace usage is not supported:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Whitespace">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Whitespace</a></p> |
| Comments      | This fault is returned when a system limit is exceeded, whether a published limit or a service-specific limit.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Applicability | All request messages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Remedy        | The client sends messages that fit the encoding limits of the service.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

5980

Table 15 – wsa:EndpointUnavailable

|               |                                                        |
|---------------|--------------------------------------------------------|
| Fault Subcode | wsa:EndpointUnavailable                                |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/addressing/fault |
| Code          | s:Receiver                                             |
| Reason        | The specified endpoint is currently unavailable.       |

|               |                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsa:RetryAfter&gt; xs:duration &lt;/wsa:RetryAfter&gt;    &lt;!-- optional --&gt;   ...optional service-specific XML content   &lt;wsman:FaultDetail&gt; A detail URI value &lt;/wsman:FaultDetail&gt; &lt;/s:Detail&gt; http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ResourceOffline Used when the resource is known, but temporarily unavailable</pre> |
| Comments      | <p>This fault is returned if the message was correct and the EPR was valid, but the specified resource is offline.</p> <p>In practice, it is difficult for a service to distinguish between "Not Found" cases and "Offline" cases. In general, wsa:DestinationUnreachable is preferable.</p>                                                                                                   |
| Applicability | All request messages                                                                                                                                                                                                                                                                                                                                                                           |
| Remedy        | The client can retry later, after the resource is again online.                                                                                                                                                                                                                                                                                                                                |

5981

**Table 16 – wsman:EventDeliverToUnusable**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:EventDeliverToUnusable                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reason        | The event source cannot process the subscription because it cannot connect to the event delivery endpoint as requested in the Delivery element.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Detail        | <pre>&lt;s:Detail&gt;   ...any service-specific content to identify the error... &lt;/s:Detail&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Comments      | <p>This fault is limited to cases of connectivity issues in contacting the “deliver to” address. These issues include:</p> <ul style="list-style-type: none"> <li>• The NotifyTo address is not usable because it is incorrect (system or device not reachable, badly formed address, and so on).</li> <li>• Permissions cannot be acquired for event delivery (for example, the wsman:Auth element does not refer to a supported security profile, and so on).</li> <li>• The credentials associated with the NotifyTo are not valid (for example, the account does not exist, the certificate thumbprint is not a hex string, and so on).</li> </ul> <p>The service can include extra information that describes the connectivity error to help in troubleshooting the connectivity problem.</p> |
| Applicability | Subscribe                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Remedy        | The client ensures connectivity from the service computer back to the event sink including firewalls and authentication/authorization configuration.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |



5982

**Table 17 – wsme:EventSourceUnableToProcess**

|               |                                                                                                                         |
|---------------|-------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:EventSourceUnableToProcess                                                                                         |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                    |
| Code          | s:Receiver                                                                                                              |
| Reason        | The event source cannot process the subscription.                                                                       |
| Detail        | None                                                                                                                    |
| Comments      | This event source is not capable of fulfilling a Subscribe request for local reasons unrelated to the specific request. |
| Applicability | Subscribe                                                                                                               |
| Remedy        | The client retries the subscription later.                                                                              |

5983

**Table 18 – wsmen:FilterDialectRequestedUnavailable**

|               |                                                                                                                                                                                                            |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:FilterDialectRequestedUnavailable                                                                                                                                                                    |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault                                                                                                                                                    |
| Code          | s:Sender                                                                                                                                                                                                   |
| Reason        | The requested filtering dialect is not supported.                                                                                                                                                          |
| Detail        | <s:Detail><br><wsmen:SupportedDialect> .... </wsmen:SupportedDialect> +<br></s:Detail>                                                                                                                     |
| Comments      | This fault is returned when the client requests a filter type or query language not supported by the service.<br>The filter dialect can vary from resource to resource or can apply to the entire service. |
| Applicability | Enumerate                                                                                                                                                                                                  |
| Remedy        | The client switches to a supported dialect or performs a simple enumeration with no filter.                                                                                                                |

5984

**Table 19 – wsme:FilteringNotSupported**

|               |                                                                                                                                                                                                                                                                                                                                                |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:FilteringNotSupported                                                                                                                                                                                                                                                                                                                     |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                                                                                                                                                                                                                                           |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                       |
| Reason        | Filtering over the event source is not supported.                                                                                                                                                                                                                                                                                              |
| Detail        | None                                                                                                                                                                                                                                                                                                                                           |
| Comments      | This fault is returned when the service does not support filtered subscriptions for the specified event source, but supports only simple delivery of all events for the resource.<br>NOTE: The service might support filtering over a different event resource or might not support filtering for <i>any</i> resource. The same fault applies. |
| Applicability | Subscribe                                                                                                                                                                                                                                                                                                                                      |
| Remedy        | The client subscribes using unfiltered delivery.                                                                                                                                                                                                                                                                                               |

5985

Table 20 – wsmen:FilteringNotSupported

|               |                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:FilteringNotSupported                                                                                                                                                                                                                                                                                                                                                                 |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault                                                                                                                                                                                                                                                                                                                                     |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                    |
| Reason        | Filtered enumeration is not supported.                                                                                                                                                                                                                                                                                                                                                      |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                                        |
| Comments      | This fault is returned when the service does not support filtering of enumerations at all, but supports only simple enumeration. If enumeration as a whole is not supported, the correct fault is wsa:ActionNotSupported.<br>NOTE: The service might support filtering over a different enumerable resource or might not support filtering for <i>any</i> resource. The same fault applies. |
| Applicability | Enumerate                                                                                                                                                                                                                                                                                                                                                                                   |
| Remedy        | The client switches to a simple enumeration.                                                                                                                                                                                                                                                                                                                                                |

5986

Table 21 – wsme:FilteringRequestedUnavailable

|               |                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:FilteringRequestedUnavailable                                                                                                                                                                                                                                                                                                                                                        |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                                                                                                                                                                                                                                                                                      |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                  |
| Reason        | The requested filter dialect is not supported.                                                                                                                                                                                                                                                                                                                                            |
| Detail        | <s:Detail><br><wsme:SupportedDialect>.. </wsme:SupportedDialect> +<br><wsman:FaultDetail> ..the following URI, if applicable </wsman:FaultDetail><br></s:Detail><br>Possible URI value:<br>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FilteringRequired                                                                                                                       |
| Comments      | This fault is returned when the client requests a filter dialect not supported by the service. In some cases, a subscription <i>requires</i> a filter because the result of an unfiltered subscription may be infinite or extremely large. In these cases, the URI http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FilteringRequired needs to be included in the s:Detail element. |
| Applicability | Subscribe                                                                                                                                                                                                                                                                                                                                                                                 |
| Remedy        | The client switches to a supported filter dialect or uses no filtering.                                                                                                                                                                                                                                                                                                                   |

5987

**Table 22 – wsman:FragmentDialectNotSupported**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:FragmentDialectNotSupported                                                                                                                                                                                                                                                                                                                                                                              |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                               |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reason        | The requested fragment filtering dialect or language is not supported.                                                                                                                                                                                                                                                                                                                                         |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsman:FragmentDialect&gt; xs:anyURI &lt;/wsman:FragmentDialect&gt;   &lt;wsman:FragmentDialect&gt; xs:anyURI &lt;/wsman:FragmentDialect&gt;   .... &lt;/s:Detail&gt;</pre> <p>The preceding optional URI values indicate supported dialects.</p>                                                                                                                                   |
| Comments      | <p>This fault is returned when the service does not support the requested fragment-level filtering dialect.</p> <p>If the implementation supports the fragment dialect in general, but not for the specific resource, this fault is still returned.</p> <p>Other resources might support the fragment dialect. This fault does not imply that the fragment dialect is not supported by the implementation.</p> |
| Applicability | Enumerate, Get, Create, Put, Delete                                                                                                                                                                                                                                                                                                                                                                            |
| Remedy        | The client uses a supported filtering dialect or no filtering.                                                                                                                                                                                                                                                                                                                                                 |

5988

**Table 23 – wsman:InternalError**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:InternalError                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Code          | s:Receiver                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Reason        | The service cannot comply with the request due to internal processing errors.                                                                                                                                                                                                                                                                                                                                                                                               |
| Detail        | <pre>&lt;s:Detail&gt;   ...service-specific extension XML elements.... &lt;/s:Detail&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                              |
| Comments      | <p>This fault is a generic error for capturing internal processing errors within the service. For example, this is the correct fault if the service cannot load necessary executable images, its configuration is corrupted, hardware is not operating properly, or any unknown or unexpected internal errors occur.</p> <p>It is expected that the service needs to be reconfigured, restarted, or reinstalled, so merely asking the client to retry will not succeed.</p> |
| Applicability | All messages                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Remedy        | The client repairs the service out-of-band to WS-Management.                                                                                                                                                                                                                                                                                                                                                                                                                |

5989

Table 24 – wsman:InvalidBookmark

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:InvalidBookmark                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Action URI    | <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/fault">http://schemas.dmtf.org/wbem/wsman/1/wsman/fault</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Reason        | The bookmark supplied with the subscription is not valid.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsman:FaultDetail&gt;     If possible, one of the following URI values   &lt;/wsman:FaultDetail&gt; &lt;/s:Detail&gt;</pre> <p>Possible URI values:</p> <p>The service is not able to back up and replay from that point:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Expired">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Expired</a></p> <p>The service is not able to decode the bookmark:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFormat">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFormat</a></p> |
| Comments      | This fault is returned if a bookmark has expired, is corrupt, or is otherwise unknown.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Applicability | Subscribe                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Remedy        | The client issues a new subscription without any bookmarks or locates the correct bookmark.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

5990

Table 25 – wsmen:InvalidEnumerationContext

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:InvalidEnumerationContext                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Action URI    | <a href="http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault">http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Code          | s:Receiver                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Reason        | The supplied enumeration context is invalid.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Comments      | <p>An invalid enumeration context was supplied with the message. Typically, this fault will happen with Pull.</p> <p>The enumeration context may be invalid due to expiration, an invalid format, or reuse of an old context no longer being tracked by the service.</p> <p>The service also can return this fault for any case where the enumerator has been terminated unilaterally on the service side, although one of the more descriptive faults is preferable, because this usually happens on out-of-memory errors (wsman:QuotaLimit), authorization failures (wsman:AccessDenied), or internal errors (wsman:InternalError).</p> |
| Applicability | Pull, Release (whether a pull-mode subscription, or a normal enumeration)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Remedy        | The client abandons the enumeration and lets the service time it out, because Release will fail as well.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

5991

**Table 26 – wsme:InvalidExpirationTime**

|               |                                                                                                                                                                                                                                                                                                                                                  |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:InvalidExpirationTime                                                                                                                                                                                                                                                                                                                       |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                                                                                                                                                                                                                                             |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                         |
| Reason        | The expiration time is not valid.                                                                                                                                                                                                                                                                                                                |
| Detail        | None                                                                                                                                                                                                                                                                                                                                             |
| Comments      | The expiration time is not valid at all or within the limits of the service.<br>This fault is used for outright errors (expirations in the past, for example) or expirations too far into the future.<br>If the service does not support expiration times at all, a wsman:UnsupportedFeature fault can be returned with the correct detail code. |
| Applicability | Subscribe                                                                                                                                                                                                                                                                                                                                        |
| Remedy        | The client issues a new subscription with a supported expiration time.                                                                                                                                                                                                                                                                           |

5992

**Table 27 – wsmen:InvalidExpirationTime**

|               |                                                                                                                                                                           |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:InvalidExpirationTime                                                                                                                                               |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault                                                                                                                   |
| Code          | s:Sender                                                                                                                                                                  |
| Reason        | The expiration time is not valid.                                                                                                                                         |
| Detail        | None                                                                                                                                                                      |
| Comments      | Because WS-Management recommends against implementing the Expiration feature, this fault might not occur with most implementations.<br>See clause 8 for more information. |
| Applicability | Enumerate                                                                                                                                                                 |
| Remedy        | Not applicable                                                                                                                                                            |

5993

**Table 28 – wsme:InvalidMessage**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:InvalidMessage                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Reason        | The request message has unknown or invalid content and cannot be processed.                                                                                                                                                                                                                                                                                                                                                                                          |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Comments      | <p>This fault is generally not used in WS-Management, although it can be used for cases not covered by other faults.</p> <p>If the content violates the schema, a wsman:SchemaValidationError fault can be sent. If specific errors occur in the subscription body, one of the more descriptive faults can be used.</p> <p>This fault is not to be used to indicate unsupported features, only unexpected or unknown content in violation of this specification.</p> |
| Applicability | Pub/sub request messages                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Remedy        | The client issues valid messages that comply with this specification.                                                                                                                                                                                                                                                                                                                                                                                                |

5994

**Table 29 – wsa:InvalidMessageInformationHeader**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsa:InvalidMessageInformationHeader                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/addressing/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Reason        | A message information header is not valid, and the message cannot be processed.                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Detail        | <pre>&lt;s:Detail&gt;   ...the invalid header... &lt;/s:Detail&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Comments      | <p>This fault can occur with any type of SOAP header error. The header might be invalid in terms of schema or value, or it might constitute a semantic error.</p> <p>This fault is not to be used to indicate an invalid resource address (a "not found" condition for the resource), but to indicate actual structural violations of the SOAP header rules in this specification.</p> <p>Examples are repeated MessageIDs, missing RelatesTo on a response, badly formed addresses, or any other missing header content.</p> |
| Applicability | All messages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Remedy        | The client reformats message using the correct format, values, and number of message information headers.                                                                                                                                                                                                                                                                                                                                                                                                                     |

5995

**Table 30 – wsman:InvalidOptions**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:InvalidOptions                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Reason        | One or more options are not valid.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Detail        | <p>&lt;s:Detail&gt;<br/>         &lt;wsman:FaultDetail&gt;<br/>             If possible, one of the following URI values<br/>         &lt;/wsman:FaultDetail&gt;<br/>         &lt;/s:Detail&gt;</p> <p>Possible URI values:<br/>         http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/NotSupported<br/>         http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidName<br/>         http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValue</p> |
| Comments      | This fault generically covers all cases where the option names or values are not valid, or they are used in incorrect combinations.                                                                                                                                                                                                                                                                                                                                               |
| Applicability | All request messages                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Remedy        | The client discovers supported option names and valid values by consulting metadata or other mechanisms. Such metadata is beyond the scope of this specification.                                                                                                                                                                                                                                                                                                                 |

5996

**Table 31 – wsman:InvalidParameter**

|               |                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:InvalidParameter                                                                                                                                                                                                                                                                                                                                                                           |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                 |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                         |
| Reason        | An operation parameter is not valid.                                                                                                                                                                                                                                                                                                                                                             |
| Detail        | <p>&lt;s:Detail&gt;<br/>         &lt;wsman:FaultDetail&gt;<br/>             If possible, one of the following URI values<br/>         &lt;/wsman:FaultDetail&gt;<br/>         &lt;/s:Detail&gt;</p> <p>Possible URI values:<br/>         http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/TypeMismatch<br/>         http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidName</p> |
| Comments      | This fault is returned when a parameter to a custom action is not valid. This fault is a default for new implementations that need to have a generic fault for this case. The method can also return any specific fault of its own.                                                                                                                                                              |
| Applicability | All messages with custom actions                                                                                                                                                                                                                                                                                                                                                                 |
| Remedy        | The client consults the WSDL for the operation and determines how to supply the correct parameter.                                                                                                                                                                                                                                                                                               |

5997

**Table 32 – wsmt:InvalidRepresentation**

|               |                                                      |
|---------------|------------------------------------------------------|
| Fault Subcode | wsmt:InvalidRepresentation                           |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/transfer/fault |
| Code          | s:Sender                                             |
| Reason        | The XML content is not valid.                        |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Detail        | <p>&lt;s:Detail&gt;<br/>                 &lt;wsman:FaultDetail&gt;<br/>                 If possible, one of the following URI values<br/>                 &lt;/wsman:FaultDetail&gt;<br/>                 &lt;/s:Detail&gt;<br/>                 Possible URI values:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValues">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValues</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MissingValues">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MissingValues</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFragment">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFragment</a></p> |
| Comments      | <p>This fault may be returned when the input XML is not valid semantically or uses the wrong schema for the resource.<br/>                 However, a wsman:SchemaValidationError fault can be returned if the error is related to XML schema violations as such, as opposed to invalid semantic values.<br/>                 Note the anomalous case in which a schema violation does not occur, but the namespace is simply the wrong one; in this case, <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace</a> is returned.</p>                                                                                                                                                                                                                                                                                               |
| Applicability | Put, Create                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Remedy        | The client corrects the request XML.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

5998

**Table 33 – wsman:InvalidSelectors**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:InvalidSelectors                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Action URI    | <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/fault">http://schemas.dmtf.org/wbem/wsman/1/wsman/fault</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Reason        | The selectors for the resource are not valid.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Detail        | <p>&lt;s:Detail&gt;<br/>                 &lt;wsman:FaultDetail&gt;<br/>                 If possible, one of the following URI values<br/>                 &lt;/wsman:FaultDetail&gt;<br/>                 &lt;/s:Detail&gt;<br/>                 Possible URI values:<br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsufficientSelectors">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsufficientSelectors</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnexpectedSelectors">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnexpectedSelectors</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/TypeMismatch">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/TypeMismatch</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValue">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValue</a><br/> <a href="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DuplicateSelectors">http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DuplicateSelectors</a></p> |
| Comments      | This fault covers all cases where the specified selectors were incorrect or unknown for the specified resource.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Applicability | All request messages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Remedy        | The client retrieves documentation or metadata and corrects the selectors.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |



5999

**Table 34 – wsa:MessageInformationHeaderRequired**

|               |                                                                                  |
|---------------|----------------------------------------------------------------------------------|
| Fault Subcode | wsa:MessageInformationHeaderRequired                                             |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/addressing/fault                           |
| Code          | s:Sender                                                                         |
| Reason        | A required header is missing.                                                    |
| Detail        | <s:Detail><br>The XML QName of the missing header<br></s:Detail>                 |
| Comments      | A required message information header (To, MessageID, or Action) is not present. |
| Applicability | All messages                                                                     |
| Remedy        | The client adds the missing message information header.                          |

6000

**Table 35 – wsman:NoAck**

|               |                                                                                                                                                                                                                            |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:NoAck                                                                                                                                                                                                                |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                           |
| Code          | s:Sender                                                                                                                                                                                                                   |
| Reason        | The receiver did not acknowledge the event delivery.                                                                                                                                                                       |
| Detail        | None                                                                                                                                                                                                                       |
| Comments      | This fault is returned when the client (subscriber) receives an event with a wsman:AckRequested header and does not (or cannot) acknowledge the receipt. The service stops sending events and terminates the subscription. |
| Applicability | Any event delivery action (including heartbeats, dropped events, and so on) in any delivery mode                                                                                                                           |
| Remedy        | For subscribers, the subscription is resubmitted without the acknowledgement option.<br>For services delivering events, the service cancels the subscription immediately.                                                  |

6001

**Table 36 – wsman:QuotaLimit**

|               |                                                                                                                           |
|---------------|---------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:QuotaLimit                                                                                                          |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                          |
| Code          | s:Sender                                                                                                                  |
| Reason        | The service is busy servicing other requests.                                                                             |
| Detail        | None                                                                                                                      |
| Comments      | This fault is returned when the SOAP message is otherwise correct, but the service has reached a resource or quota limit. |
| Applicability | All messages                                                                                                              |
| Remedy        | The client can retry later.                                                                                               |

6002

**Table 37 – wsman:SchemaValidationError**

|               |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:SchemaValidationError                                                                                                                                                                                                                                                                                                                                                                                        |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                   |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                           |
| Reason        | The supplied SOAP violates the corresponding XML schema definition.                                                                                                                                                                                                                                                                                                                                                |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                                                               |
| Comments      | This fault is used for any XML parsing failure or schema violations.<br>Full validation of the SOAP against schemas is not expected in real-time, but processors might in fact notice schema violations, such as type mismatches. In all of these cases, this fault applies.<br>In debugging modes where validation is occurring, this fault can be returned for <i>all</i> errors noted by the validating parser. |
| Applicability | All messages                                                                                                                                                                                                                                                                                                                                                                                                       |
| Remedy        | The client corrects the message.                                                                                                                                                                                                                                                                                                                                                                                   |

6003

**Table 38 – wsmen:TimedOut**

|               |                                                                                                                        |
|---------------|------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:TimedOut                                                                                                         |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault                                                                |
| Code          | s:Receiver                                                                                                             |
| Reason        | The enumerator has timed out and is no longer valid.                                                                   |
| Detail        | None                                                                                                                   |
| Comments      | This fault is not to be used in WS-Management due to overlap with wsman:TimedOut, which covers all the other messages. |
| Applicability | Pull                                                                                                                   |
| Remedy        | The client can retry the Pull request.                                                                                 |

6004

**Table 39 – wsman:TimedOut**

|               |                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:TimedOut                                                                                                                                                                                                                                                                                                                                                                   |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                 |
| Code          | s:Receiver                                                                                                                                                                                                                                                                                                                                                                       |
| Reason        | The operation has timed out.                                                                                                                                                                                                                                                                                                                                                     |
| Detail        | None                                                                                                                                                                                                                                                                                                                                                                             |
| Comments      | The operation could not be completed within the wsman:OperationTimeout value, or an internal override timeout was reached by the service while trying to process the request.<br>This fault is also returned in all enumerations when no content is available for the current Pull request. Clients can simply retry the Pull request again until a different fault is returned. |
| Applicability | All requests                                                                                                                                                                                                                                                                                                                                                                     |
| Remedy        | The client can retry the operation.<br>If the operation is a write (delete, create, or custom operation), the client can consult the system operation log before blindly attempting a retry or attempt a Get or other read operation to try to discover the result of the previous operation.                                                                                    |

6005

**Table 40 – wsme:UnableToRenew**

|               |                                                                                                      |
|---------------|------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:UnableToRenew                                                                                   |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                 |
| Code          | s:Sender                                                                                             |
| Reason        | The subscription could not be renewed.                                                               |
| Detail        | None                                                                                                 |
| Comments      | This fault is returned in all cases where the subscription cannot be renewed but is otherwise valid. |
| Applicability | wsme:Renew                                                                                           |
| Remedy        | The client issues a new subscription.                                                                |

6006

**Table 41 – wsme:UnsupportedExpirationType**

|               |                                                                                                                                                                                                     |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:UnsupportedExpirationType                                                                                                                                                                      |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/eventing/fault                                                                                                                                                |
| Code          | s:Sender                                                                                                                                                                                            |
| Reason        | The specified expiration type is not supported.                                                                                                                                                     |
| Detail        | None                                                                                                                                                                                                |
| Comments      | A specific time for expiration (as opposed to duration) is not supported. This fault is not to be used if the value itself is incorrect; it is only to be used if the <i>type</i> is not supported. |
| Applicability | Subscribe                                                                                                                                                                                           |
| Remedy        | The client corrects the expiration to use a duration time.                                                                                                                                          |

6007

**Table 42 – wsmen:UnsupportedExpirationType**

|               |                                                                                                                                                                                                                                                                                                       |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:UnsupportedExpirationType                                                                                                                                                                                                                                                                       |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/09/enumeration/fault                                                                                                                                                                                                                                               |
| Code          | s:Sender                                                                                                                                                                                                                                                                                              |
| Reason        | The specified expiration type is not supported.                                                                                                                                                                                                                                                       |
| Detail        | None                                                                                                                                                                                                                                                                                                  |
| Comments      | The specified expiration type is not supported. For example, a specific time-based expiration type might not be supported (as opposed to a duration-based expiration type). This fault is not to be used if the value itself is incorrect; it is only to be used if the <i>type</i> is not supported. |
| Applicability | Enumerate                                                                                                                                                                                                                                                                                             |
| Remedy        | The client corrects the expiration time or omits it and retries.                                                                                                                                                                                                                                      |

6008

Table 43 – wsman:UnsupportedFeature

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsman:UnsupportedFeature                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Action URI    | http://schemas.dmtf.org/wbem/wsman/1/wsman/fault                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Code          | s:Sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Reason        | The specified feature is not supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Detail        | <pre>&lt;s:Detail&gt;   &lt;wsman:FaultDetail&gt;     If possible, one of the following URI values   &lt;/wsman:FaultDetail&gt; &lt;/s:Detail&gt;</pre> <p>Possible URI values:</p> <ul style="list-style-type: none"> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Ack</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AsynchronousRequest</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Bookmarks</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DeliveryRetries</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EnumerationMode</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ExpirationTime</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FilteringRequired</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FormatMismatch</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FragmentLevelAccess</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Heartbeats</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsecureAddress</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Locale</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxElements</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopePolicy</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxTime</li> <li>http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OperationTimeout</li> </ul> |
| Comments      | This fault indicates that an unsupported feature was attempted.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Applicability | Any message                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Remedy        | The client corrects or removes the unsupported feature request and retries.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

6009

Table 44 – wsme:UnsupportedExpirationType

|               |                                                                                                                                                                                                                           |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:UnsupportedExpirationType                                                                                                                                                                                            |
| Action URI    | http://schemas.xmlsoap.org/ws/2004/08/addressing/fault                                                                                                                                                                    |
| Code          | s:Sender                                                                                                                                                                                                                  |
| Reason        | Only expiration durations are supported.                                                                                                                                                                                  |
| Detail        | None                                                                                                                                                                                                                      |
| Comments      | This fault is sent when a Subscribe request specifies an expiration time and the event source is only capable of accepting expiration durations; for instance, if the event source does not have access to absolute time. |
| Applicability | Subscribe, wsme:Renew                                                                                                                                                                                                     |

|        |  |
|--------|--|
| Remedy |  |
|--------|--|

6010

**Table 45 – wsmen:UnableToRenew**

|               |                                                                                                                                            |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsmen:UnableToRenew                                                                                                                        |
| Action URI    | <a href="http://schemas.xmlsoap.org/ws/2004/08/addressing/fault">http://schemas.xmlsoap.org/ws/2004/08/addressing/fault</a>                |
| Code          | s:Sender                                                                                                                                   |
| Reason        | <i>Text explaining the failure; e.g., "The event source has too many subscribers".</i>                                                     |
| Detail        | None                                                                                                                                       |
| Comments      | This fault is sent when the event source is not capable of fulfilling a Renew request for local reasons unrelated to the specific request. |
| Applicability | wsmen:Renew                                                                                                                                |
| Remedy        |                                                                                                                                            |

6011

**Table 46 – wsa:InvalidMessage**

|               |                                                                                                                                                                                                                              |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsa:InvalidMessage                                                                                                                                                                                                           |
| Action URI    | <a href="http://schemas.xmlsoap.org/ws/2004/08/addressing/fault">http://schemas.xmlsoap.org/ws/2004/08/addressing/fault</a>                                                                                                  |
| Code          | s:Sender                                                                                                                                                                                                                     |
| Reason        | <i>The message is not valid and cannot be processed.</i>                                                                                                                                                                     |
| Detail        | <i>The invalid message</i>                                                                                                                                                                                                   |
| Comments      | If a request message does not comply with the corresponding outline in the previous row, the request shall fail and the event source or subscription manager may generate this fault indicating that the request is invalid. |
| Applicability | Subscribe, Renew, wsme:GetStatus, Unsubscribe                                                                                                                                                                                |
| Remedy        |                                                                                                                                                                                                                              |

6012

**Table 47 – wsme:CannotProcessFilter**

|               |                                                                                                                             |
|---------------|-----------------------------------------------------------------------------------------------------------------------------|
| Fault Subcode | wsme:CannotProcessFilter                                                                                                    |
| Action URI    | <a href="http://schemas.xmlsoap.org/ws/2004/08/addressing/fault">http://schemas.xmlsoap.org/ws/2004/08/addressing/fault</a> |
| Code          | s:Sender                                                                                                                    |
| Reason        | <i>Cannot filter as requested</i>                                                                                           |
| Detail        | None                                                                                                                        |
| Comments      | A filter was specified can not be processed.                                                                                |
| Applicability | Subscribe                                                                                                                   |
| Remedy        |                                                                                                                             |

6013

6014  
6015  
6016  
6017

## ANNEX A (informative)

### Notational Conventions

- 6018 This annex specifies the notations and namespaces used in this specification.
- 6019 This specification uses the following syntax to define normative outlines for messages:
- 6020 • The syntax appears as an XML instance, but values in italics indicate data types instead of  
6021 values.
  - 6022 • Characters are appended to elements and attributes to indicate cardinality:  
6023 "?" (0 or 1)  
6024 "\*" (0 or more)  
6025 "+" (1 or more)
  - 6026 • The character "|" indicates a choice between alternatives.
  - 6027 • The characters "[" and "]" indicate that enclosed items are to be treated as a group with  
6028 respect to cardinality or choice.
  - 6029 • An ellipsis ("...") indicates a point of extensibility that allows other child or attribute content.  
6030 Additional children and attributes may be added at the indicated extension points but must  
6031 not contradict the semantics of the parent or owner, respectively. If a receiver does not  
6032 recognize an extension, the receiver should not process the message and may fault.
  - 6033 • XML namespace prefixes (see Table A-1) indicate the namespace of the element being  
6034 defined.
- 6035 Throughout the document, whitespace within XML element values is used for readability. In practice,  
6036 a service can accept and strip leading and trailing whitespace within element values as if whitespace  
6037 had not been used.

#### 6038 **A.1 XML Namespaces**

6039 Table A-1 lists XML namespaces used in this specification. The choice of any namespace prefix is  
6040 arbitrary and not semantically significant. Unless otherwise noted, the XML Schema for each  
6041 specification can be retrieved by resolving the XML namespace URI for each specification listed in  
6042 Table A-1.

6043

Table A-1 – Prefixes and XML Namespaces Used in This Specification

| Prefix | XML Namespace                                                   | Specification                                                 |
|--------|-----------------------------------------------------------------|---------------------------------------------------------------|
| wsman  | http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd                  | This specification                                            |
| wsmid  | http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd | This specification – discovery of supported protocol versions |
| s      | http://www.w3.org/2003/05/soap-envelope                         | <u>SOAP 1.2</u>                                               |
| xs     | http://www.w3.org/2001/XMLSchema                                | <u>XML Schema 1, XML Schema 2</u>                             |
| wsdl   | http://schemas.xmlsoap.org/wsdl                                 | WSDL/1.1                                                      |
| wsa    | Either wsa04 or wsa10                                           | Either wsa04 or wsa10                                         |
| wsa04  | http://schemas.xmlsoap.org/ws/2004/08/addressing                | Clause 5 of this specification                                |
| wsa10  | http://www.w3.org/2005/08/addressing                            | <u>WS-Addressing W3C Recommendation</u>                       |
| wsam   | http://www.w3.org/2007/05/addressing/metadata                   | <u>WS-Addressing Metadata W3C Recommendation</u>              |
| wsme   | http://schemas.xmlsoap.org/ws/2004/08/eventing                  | Clause 10 of this specification                               |
| wsmen  | http://schemas.xmlsoap.org/ws/2004/09/enumeration               | Clause 8 of this specification                                |
| wsmt   | http://schemas.xmlsoap.org/ws/2004/09/transfer                  | Clause 7 of this specification                                |
| wsp    | http://schemas.xmlsoap.org/ws/2004/09/policy                    | <u>WS-Policy</u>                                              |

6044

6045  
6046  
6047  
6048

## ANNEX B (normative)

### Conformance

6049 This annex specifies the conformance rules used in this specification.

6050 An implementation is not conformant with this specification if it fails to satisfy one or more of the  
6051 “shall” or “required” level requirements defined in the conformance rules for each section, as indicated  
6052 by the following format:

6053 **Rnnnn:** Rule text

6054 General conformance rules are defined as follows:

6055 **RB-1:** To be conformant, the service shall comply with all the rules defined in this  
6056 specification. Items marked with shall are required, and items marked with should are highly  
6057 advised to maximize interoperation. Items marked with may indicate the preferred implementation  
6058 for expected features, but interoperation is not affected if they are ignored.

6059 **RB-2:** Conformant services of this specification shall use this XML namespace Universal  
6060 Resource Identifier:

6061 (1) <http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd>

6062 **RB-3:** A SOAP node shall not use the XML namespace identifier for this specification unless it  
6063 complies with the conformance rules in this specification.

6064 This specification does not mandate that all messages and operations need to be supported. It only  
6065 requires that any supported message or operation obey the conformance rules for that message or  
6066 operation. It is important that services not use the XML namespace identifier for WS-Management in  
6067 SOAP operations in a manner that is inconsistent with the rules defined in this specification.



6068  
6069  
6070  
6071

## ANNEX C (normative)

### HTTP(S) Transport and Security Profile

#### 6072 C.1 General

6073 Although WS-Management is a SOAP protocol and not tied to a specific network transport,  
6074 interoperation requires some common standards to be established. This clause centers on  
6075 establishing common usage over HTTP 1.1 and HTTPS. In addition to HTTP and HTTPS, this  
6076 specification allows any SOAP-enabled transport to be used as a carrier for WS-Management  
6077 messages.

6078 For identification and referencing, each transport is identified by a URI, and each authentication  
6079 mechanism defined in this specification is also identified by a URI.

6080 As new transports are standardized, they can also acquire a URI for referencing purposes, and any  
6081 new authentication mechanisms that they expose can also be assigned URIs for publication and  
6082 identification purposes in XML documents. As new transports are standardized for WS-Management,  
6083 the associated transport-specific requirements can be defined and published to ensure  
6084 interoperability.

6085 The SOAP HTTP binding described in section 7 of [SOAP Version 1.2 Part 2: Adjuncts](#) is  
6086 used for WS-Management encoding over HTTP and HTTPS.

#### 6087 C.2 HTTP(S) Binding

6088 This clause clarifies how SOAP messages are bound to HTTP(S).

6089 **RC.2-1:** A service that supports the SOAP HTTP(S) binding shall at least support it using  
6090 HTTP 1.1.

6091 **RC.2-2:** A service shall at least implement the Responding SOAP Node of the SOAP  
6092 Request-Response Message Exchange Pattern:

6093 <http://www.w3.org/2003/05/soap/mep/request-response/>

6094 **RC.2-3:** A service may choose not to implement the Responding SOAP Node of the SOAP  
6095 Response Message Exchange Pattern:

6096 <http://www.w3.org/2003/05/soap/mep/soap-response/>

6097 **RC.2-4:** A service may choose not to support the SOAP Web Method Feature.

6098 **RC.2-5:** A service shall at least implement the Responding SOAP Node of an HTTP one-way  
6099 Message Exchange Pattern where the SOAP Envelope is carried in the HTTP Request and the  
6100 HTTP Response has a Status Code of 202 Accepted and an empty Entity Body (no SOAP  
6101 Envelope).

6102 The message exchange pattern described in RB.2-5 is used to carry SOAP messages that  
6103 require no response.

6104 **RC.2-6:** A service shall at least support Request Message SOAP Envelopes and one-way  
6105 SOAP Envelopes delivered using HTTP Post.

- 6106 **RC.2-7:** In cases where the service cannot respond with a SOAP message, the HTTP error  
6107 code 500 (Internal Server Error) should be returned and the client side should close the  
6108 connection.
- 6109 **RC.2-8:** For services that support HTTPS, the transport layer handles negotiation of the  
6110 proper encryption protocol. Services may implement an Identify response that is unauthenticated  
6111 to facilitate negotiation. **RC.2-9:** When delivering faults, an HTTP status code of 500  
6112 should be used in the response for s:Receiver faults, and a code of 400 should be used for  
6113 s:Sender faults.
- 6114 **RC.2-10:** The URL used with the HTTP-Post operation to deliver the SOAP message is not  
6115 required to have the same content as the wsa:To URI used in the SOAP address. Often, the  
6116 HTTP URL has the same content as the wsa:To URI in the message, but may additionally contain  
6117 other message routing fields suffixed to the network address using a service-defined separator  
6118 token sequence. It is recommended that services require only the wsa:To network address URL  
6119 to promote uniform client-side processing and behavior, and to include service-level routing in  
6120 other parts of the address.
- 6121 **RC.2-11:** In the absence of other requirements, it is recommended that the path portion of the  
6122 URL used with the HTTP-POST operation be /wsman for resources that require authentication  
6123 and /wsman-anon for resources that do not require authentication. If these paths are used,  
6124 unauthenticated requests should not be supported for /wsman and authentication must not be  
6125 required for /wsman-anon.
- 6126 **RC.2-12:** If the SOAPAction header is present in an HTTP/HTTPS-based request that carries a  
6127 SOAP message, it must match the wsa:Action URI present in the SOAP message. The  
6128 SOAPAction header is optional, and a service must not fault a request if this header is missing.
- 6129 Because WS-Management is based on SOAP 1.2, the optional SOAPAction header is merely  
6130 used as an optimization. If present, it shall match the wsa:Action URI used in the SOAP  
6131 message. The service is permitted to fault the request by simply examining the SOAPAction  
6132 header, if the action is not valid, without examining the SOAP content. However, the service may  
6133 not fault the request if the SOAPAction header is omitted.
- 6134 **RC.2-13:** If a service supports attachments, the service shall support the HTTP Transmission  
6135 Optimization Feature.
- 6136 **RC.2-14:** If a service cannot process a message with an attachment or unsupported encoding  
6137 type, and the transport is HTTP or HTTPS, it shall return HTTP error 415 as its response  
6138 (unsupported media).
- 6139 **RC.2-15:** If a service cannot process a message with an attachment or unsupported encoding  
6140 type using transports other than HTTP/HTTPS, it should return a wsman:EncodingLimit fault with  
6141 the following detail code:
- 6142 `http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EncodingType`

### 6143 C.3 HTTP(S) Security Profiles

6144 This specification defines a set of security profiles for use with HTTP and HTTPS. Conformant  
6145 services need not support HTTP or HTTPS, but if supported these predefined profiles provide the  
6146 client with at least one way to access the service. Other specifications can define additional profiles  
6147 for use with HTTP or HTTPS.

- 6148 **RC.3-1:** A conformant service that supports HTTP shall support one of the predefined HTTP-  
6149 based profiles.

6150 **RC.3-2:** A conformant service that supports HTTPS shall support one of the predefined  
 6151 HTTPS-based profiles.

6152 **RC.3-3:** A conformant service should not expose WS-Management over a completely  
 6153 unauthenticated HTTP channel except for situations such as Identify (see clause 11), debugging,  
 6154 or as determined by the service.

6155 The service is not required to export only a single HTTP or HTTPS address. The service can export  
 6156 multiple addresses, each of which supports a specific security profile or multiple profiles.

6157 If clients support all predefined profiles, they are assured of some form of secure access to a  
 6158 WS-Management implementation that supports HTTP, HTTPS, or both.

6159 **C.3.1 <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/basic>**

6160 This profile is essentially the "standard" profile, but it is limited to Basic authentication.

6161 The typical sequence is shown in Table C-1.

6162 **Table C-1 – Basic Authentication Sequence**

|   | Client                                        |   | Service                                                                 |
|---|-----------------------------------------------|---|-------------------------------------------------------------------------|
| 1 | Client connects with no authorization header. | ➔ | Service sees no header.                                                 |
| 2 |                                               | ➔ | Service sends 401 return code, listing Basic as the authorization mode. |
| 3 | Client provides Basic authorization header.   | ➔ | Service authenticates the client.                                       |

6163 This behavior is normal for HTTP. If the client connects with a Basic authorization header initially and  
 6164 if it is valid, the request immediately succeeds.

6165 Basic authentication is not recommended for unsecured transports. If used with HTTP alone, for  
 6166 example, the transmission of the password constitutes a security risk. However, if the HTTP transport  
 6167 is secured with IPSec, for example, the risk is substantially reduced.

6168 Similarly, Basic authentication is suitable when performing testing, prototyping, or diagnosis.

6169 **C.3.2 <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/digest>**

6170 This profile is essentially the same as the "standard" profile, but it is limited to the use of Digest  
6171 authentication.

6172 The typical sequence is shown in Table C-2.

6173 **Table C-2 – Digest Authentication Sequence**

|   | Client                                        |   | Service                                                                  |
|---|-----------------------------------------------|---|--------------------------------------------------------------------------|
| 1 | Client connects with no authorization header. | → | Service sees no header.                                                  |
| 2 |                                               | ← | Service sends 401 return code, listing Digest as the authorization mode. |
| 3 | Client provides Digest authorization header.  | → |                                                                          |
| 4 |                                               | ← | Service begins authorization sequence of secure token exchange.          |
| 5 | Client continues authorization sequence.      | → | Service authenticates client.                                            |

6174 This behavior is normal for HTTP. If the client connects with a Digest authorization header initially and  
6175 if it is valid, the token exchange sequence begins.

6176 **C.3.3 <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/basic>**

6177 This profile establishes the use of Basic authentication over HTTPS. This profile is used when only a  
6178 server-side certificate encrypts the connection, but the service still needs to authenticate the client.

6179 The typical sequence is shown in Table C-3.

6180 **Table C-3 – Basic Authentication over HTTPS Sequence**

|   | Client                                                    |   | Service                                                                 |
|---|-----------------------------------------------------------|---|-------------------------------------------------------------------------|
| 1 | Client connects with no authorization header using HTTPS. | → | Service sees no header, but establishes an encrypted connection.        |
| 2 |                                                           | ← | Service sends 401 return code, listing Basic as the authorization mode. |
| 3 | Client provides Basic authorization header.               | → | Service authenticates the client.                                       |

6181 If the client connects with a Basic authorization header initially and if it is valid, the request  
6182 immediately succeeds.

6183 **C.3.4 <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/digest>**

6184 This profile establishes the use of Digest authentication over HTTPS. This profile is used when only a  
6185 server-side certificate encrypts the connection, but the service still needs to authenticate the client.

6186 The typical sequence is shown in Table C-4.

6187 **Table C-4 – Digest Authentication over HTTPS Sequence**

|   | Client                                                    |   | Service                                                          |
|---|-----------------------------------------------------------|---|------------------------------------------------------------------|
| 1 | Client connects with no authorization header using HTTPS. | → | Service sees no header, but establishes an encrypted connection. |

|   |                                              |   |                                                                 |
|---|----------------------------------------------|---|-----------------------------------------------------------------|
| 2 |                                              | ← | Service sends 401 return code, listing Digest as the auth mode. |
| 3 | Client provides Digest authorization header. | → |                                                                 |
| 4 |                                              | ← | Service begins authorization sequence of secure token exchange. |
| 5 | Client continues authorization sequence.     | → | Service authenticates client.                                   |

6188 This behavior is normal for HTTPS. If the client connects with a Digest authorization header initially  
 6189 and if it is valid, the token exchange sequence begins.

6190 **C.3.5 http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/  
 6191 mutual**

6192 In this security mode, the client supplies an X.509 certificate that is used to authenticate the client. No  
 6193 HTTP or HTTPS authorization header is required in the HTTP-Post request.

6194 However, as a hint to the service, the following HTTP/HTTPS authorization header may be present.

6195 Authorization: http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual

6196 Because the service can be configured to always look for the certificate, this authorization header is  
 6197 not required.

6198 This simple sequence is shown in Table C-5.

6199 **Table C-5 – HTTPS with Client Certificate Sequence**

|   | Client                                                                          |   | Service                                                                             |
|---|---------------------------------------------------------------------------------|---|-------------------------------------------------------------------------------------|
| 1 | Client connects with no authorization header but supplies an X.509 certificate. | → | Service ignores the authorization header and retrieves the client-side certificate. |
| 2 |                                                                                 | ← | Service accepts or denies access with 403.7 or 403.16 return codes.                 |

6200 **C.3.6 http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/  
 6201 mutual/basic**

6202 In this profile, the http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual profile is  
 6203 used first to authenticate both sides using X.509 certificates. Individual operations are subsequently  
 6204 authenticated using HTTP Basic authorization headers.

6205 This profile authenticates both the client and service initially and provides one level of security,  
 6206 typically at the machine or device level. The second level of authentication typically performs  
 6207 authorization for specific operations, although it can act as a simple, secondary authentication  
 6208 mechanism with no authorization semantics.

6209 The typical sequence is shown in Table C-6.

6210 **Table C-6 – Basic Authentication over HTTPS with Client Certificate Sequence**

|   | Client                                                             |   | Service                                                                                                                                                    |
|---|--------------------------------------------------------------------|---|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Client connects with certificate and special authorization header. | → | Service queries for client certificate and authenticates. If certificate is missing or invalid, the sequence stops here with 403.7 or 403.16 return codes. |

|   |                                                                                                                             |   |                                                                                                                                       |
|---|-----------------------------------------------------------------------------------------------------------------------------|---|---------------------------------------------------------------------------------------------------------------------------------------|
| 2 |                                                                                                                             | ← | After authenticating the certificate, the service sends 401 return code, listing available Basic authorization mode as a requirement. |
| 3 | Client selects Basic as the authorization mode to use and includes it in the Authorization header, as defined for HTTP 1.1. | → | Service authenticates the client again before performing the operation.                                                               |

6211 In the initial request, the HTTPS authorization header must be as follows:

6212 Authorization: http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/basic

6213 This indicates to the service that this special mode is in use, and that it can query for the client  
6214 certificate to ensure that subsequent requests are properly challenged for Basic authorization if the  
6215 HTTP Authorization header is missing from a request.

6216 The Authorization header is treated as normal HTTP basic:

6217 Authorization: Basic ...user/password encoding

6218 This use of Basic authentication is secure (unlike its normal use in HTTP) because the transmission  
6219 of the user name and password is performed over an encrypted connection.

6220 **C.3.7 http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/  
6221 mutual/digest**

6222 This profile is the same as  
6223 http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/basic, except that the HTTP  
6224 Digest authentication model is used after the initial X.509 certificate-based mutual authentication is  
6225 completed.

6226 In the initial request, the HTTPS authorization header must be as follows:

6227 Authorization:  
6228 http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/digest

6229 **C.3.8 http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/  
6230 spnego-kerberos**

6231 In this profile, the client connects to the server using HTTPS with only server-side certificates to  
6232 encrypt the connection.

6233 Authentication is carried out based on [RFC 4559](#), which describes the use of GSSAPI SPNEGO over  
6234 HTTP (Table C-7). This mechanism allows HTTP to carry out the negotiation protocol of [RFC 4178](#) to  
6235 authenticate the user based on Kerberos Version 5.

6236 **Table C-7 – SPNEGO Authentication over HTTPS Sequence**

|   | Client                                                                                             |   | Service                                                                                                |
|---|----------------------------------------------------------------------------------------------------|---|--------------------------------------------------------------------------------------------------------|
| 1 | Client connects with no authorization header using HTTPS.                                          | → | Service sees no header, but establishes an encrypted connection.                                       |
| 2 |                                                                                                    | ← | Service sends 401 return code, listing <b>Negotiate</b> as an available HTTP authentication mechanism. |
| 3 | Client uses the referenced Internet draft to start a SPNEGO sequence to negotiate for Kerberos V5. | → | ...                                                                                                    |

|   |                          |   |                                                                              |
|---|--------------------------|---|------------------------------------------------------------------------------|
| 4 | ...                      | ← | Service engages in SPNEGO sequence to authenticate client using Kerberos V5. |
| 5 | Client is authenticated. | → | Service authenticates client.                                                |

6237 **C.3.9 [http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/  
mutual/spnego-kerberos](http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/spnego-kerberos)**

6239 This mode is the same as <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/spnego-kerberos> except that the server and client mutually authenticate one another at the transport layer prior to beginning the Kerberos authentication sequence (Table C-8). See [RFC 4178](#) for details.

6242 **Table C-8 – SPNEGO Authentication over HTTPS with Client Certificate Sequence**

|   | Client                                                                                             |   | Service                                                                                                                                                      |
|---|----------------------------------------------------------------------------------------------------|---|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Client connects with no authorization header using HTTPS.                                          | → | Service queries for client certificate and authenticates. If certificate is missing or invalid, the sequence stops here with 403.7 or 403.16 return codes.   |
| 2 |                                                                                                    | ← | After the mutual certificate authentication sequence, service sends 401 return code, listing <b>Negotiate</b> as an available HTTP authentication mechanism. |
| 3 | Client uses the referenced Internet draft to start a SPNEGO sequence to negotiate for Kerberos V5. | → | ...                                                                                                                                                          |
| 4 | ...                                                                                                | ← | Service engages in SPNEGO sequence to authenticate client using Kerberos V5.                                                                                 |
| 5 | Client is authenticated.                                                                           | → | Service authenticates client.                                                                                                                                |

6243 Typically, this is used to mutually authenticate devices or machines, and then subsequently perform user- or role-based authentication.

6245 **C.3.10 <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/spnego-kerberos>**

6247 This profile is the same as <http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/spnego-kerberos> except that it is performed over an HTTP connection. See [RFC 4178](#) for details.

6249 Although this profile supports secure authentication, because it is not encrypted, it represents security risks such as information disclosure because the SOAP traffic is in plain text. It is not to be used in environments that require a high level of security.

6252 **C.4 IPSec and HTTP**

6253 HTTP with Basic authentication is weak on an unsecured network. If IPSec is in use, however, this weakness is no longer an issue. IPSec provides high-quality cryptographic security, data origin authentication, and anti-replay services.

6256 Because IPSec is intended for machine-level authentication and network traffic protection, it is insufficient for real-world management in many cases, which can require additional authentication of specific users to authorize access to resource classes and instances. IPSec needs to be used in conjunction with one of the profiles in this clause for user-level authentication. However, it obviates the need for HTTPS-based traffic and allows safe use of HTTP-based profiles.

6261 From the network perspective, the use of HTTP Basic authentication when the traffic is carried over a  
6262 network secured by IPSec is intrinsically safe and equivalent to using HTTPS with server-side  
6263 certificates. Other specifications can define IPSec security profiles that combine IPSec  
6264 with appropriate authentication mechanisms.



6265  
6266  
6267  
6268

## ANNEX D (informative)

### XPath Support

#### 6269 D.1 General

6270 Implementations typically need to support XPath for several purposes, such as fragment-level access  
6271 (7.7), datasets (8), and filtering (10.2.2). Because the full [XPath 1.0](#) specification is large, subsets are  
6272 typically required in resource-constrained implementations.

6273 The purpose of this clause is to identify the minimum set of syntactic elements that implementations  
6274 can provide to promote maximum interoperability. In most cases, implementations provide large  
6275 subsets of full XPath, but they need additional definitions to ensure that the subsets meet minimum  
6276 requirements. The Level 1 and Level 2 BNF definitions in this annex establish such minimums for use  
6277 in the WS-Management space.

6278 This specification defines two subset profiles for XPath: Level 1 with basic node selector support and  
6279 no filtering (for supporting Fragment-level access as described in 7.7), and Level 2 with basic filtering  
6280 support (for enumerating and receiving notifications). Level 2 is a formal superset of Level 1.

6281 The following BNFs both are formal LL(1) grammars. A parser can be constructed automatically from  
6282 the BNF using an appropriate tool, or a recursive-descent parser can be implemented manually by  
6283 inspection of the grammar.

6284 Within the grammars, non-terminal tokens are surrounded by angled brackets, and terminal tokens  
6285 are in uppercase and not surrounded by angled brackets.

6286 XML namespace support is explicitly absent from these definitions. Processors that meet the syntax  
6287 requirements can provide a mode in which the elements are processed without regard to XML  
6288 namespaces, but can also provide more powerful, namespace-aware processing.

6289 The default execution context of the XPath is specified explicitly in 8.4 and 10.2.2.

6290 For the following dialects, XML namespaces and QNames are not expected to be supported by  
6291 default and can be silently ignored by the implementation.

6292 These dialects are for informational purposes only and are not intended as Filter Dialects in actual  
6293 SOAP messages. Because they are XPath compliant (albeit subsets), the Filter Dialect in the SOAP  
6294 messages is still that of full XPath:

6295 <http://www.w3.org/TR/1999/REC-xpath-19991116>

6296 **D.2 Level 1**

6297 Level 1 contains just the necessary XPath to identify nodes within an XML document or fragment and  
 6298 is targeted for use with Fragment-level access (7.7) of this specification.

6299 EXAMPLE:

```

6300 (1) <path> ::= <root_selector> TOKEN_END_OF_INPUT;
6301 (2) <root_selector> ::= TOKEN_SLASH <element_sequence>;
6302 (3) <root_selector> ::= <attribute>;
6303 (4) <root_selector> ::= <relpath> <element_sequence>;
6304 (5) <root_selector> ::= TOKEN_DOT
6305 (6) <relpath> ::= <>;
6306 (7) <relpath> ::= TOKEN_DOT TOKEN_SLASH;
6307 (8) <relpath> ::= TOKEN_DOT_DOT TOKEN_SLASH;
6308 (9) <element_sequence> ::= <element> <optional_filter_expression> <more>;
6309 (10) <more> ::= TOKEN_SLASH <follower>;
6310 (11) <more> ::= <>;
6311 (12) <follower> ::= <attribute>;
6312 (13) <follower> ::= <text_function>;
6313 (14) <follower> ::= <element_sequence>;
6314 (15) <optional_filter_expression> ::=
6315 (16) TOKEN_OPEN_BRACKET <filter_expression> TOKEN_CLOSE_BRACKET;
6316 (17) <optional_filter_expression> ::= <>;
6317 (18) <attribute> ::= TOKEN_AT_SYMBOL <name>;
6318 (19) <element> ::= <name>;
6319 (20) <text_function> ::=
6320 (21) TOKEN_TEXT TOKEN_OPEN_PAREN TOKEN_CLOSE_PAREN;
6321 (22) <name> ::= TOKEN_XML_NAME;
6322 (23) <filter_expression> ::= <array_location>;
6323 (24) <array_location> ::= TOKEN_UNSIGNED_POSITIVE_INTEGER;

```

6324 This dialect allows selecting any XML node based on its name or array position, or any attribute by its  
 6325 name. Optionally, the text() NodeTest can trail the entire expression to select only the raw value of  
 6326 the name, excluding the XML element name wrapper.

6327 Terminals in the grammar are defined as shown in Table D-1.

6328 **Table D-1 – XPath Level 1 Terminals**

|                                 |                                        |
|---------------------------------|----------------------------------------|
| TOKEN_SLASH                     | The character '/'                      |
| TOKEN_DOT                       | The character '.'                      |
| TOKEN_DOT_DOT                   | The characters '..'                    |
| TOKEN_END_OF_INPUT              | End of input                           |
| TOKEN_OPEN_BRACKET              | The character '['                      |
| TOKEN_CLOSE_BRACKET             | The character ']'                      |
| TOKEN_AT_SYMBOL                 | The character '@'                      |
| TOKEN_XML_NAME                  | Equivalent to XML Schema type xs:token |
| TOKEN_UNSIGNED_POSITIVE_INTEGER | Values in the subrange 1..4294967295   |
| TOKEN_TEXT                      | The characters 'text'                  |
| TOKEN_OPEN_PAREN                | The character '('                      |
| TOKEN_CLOSE_PAREN               | The character ')'                      |

6329 Using the following XML fragment, some examples are shown assuming that the element "a" is the  
6330 context node (that is, represents the resource or event document).

6331 **EXAMPLE 1:**

```
6332 (1) <Envelope>
6333 (2) <Body>
6334 (3) <a>
6335 (4) <b x="y"> 100
6336 (5) <c>
6337 (6) <d> 200 </d>
6338 (7) </c>
6339 (8) <c>
6340 (9) <d> 300 </d>
6341 (10) <d> 400 </d>
6342 (11) </c>
6343 (12)
6344 (13) </Body>
6345 (14) </Envelope>
```

6346 **EXAMPLE 2:**

```
6347 (1) / // Selects <a> and all its content
6348 (2) /a // Selects <a> and all its content
6349 (3) . // Selects <a> and all its content
6350 (4) ../a // Selects <a> and all its content
6351 (5) b // Selects <b x="y"> 100
6352 (6) c // Selects both <c> nodes, one after the other
6353 (7) c[1] // Selects <c><d>200</d></c>
6354 (8) c[2]/d[2] // Selects <d> 400 </d>
6355 (9) c[2]/d[2]/text() // Selects 400
6356 (10) b/text() // Selects 100
6357 (11) b/@x // Selects x="y"
```

6358 The only filtering expression capability is an array selection. XPath can return a node set. In 7.7 of  
6359 this specification, the intent is to select a specific node, not a set of nodes, so if the situation occurs  
6360 as illustrated on line (20) above, most implementations simply return a fault stating that it is unclear  
6361 which <c> was meant and require the client to actually select one of the two available <c> elements  
6362 using the array syntax. Also, text() cannot be suffixed to attribute selection.

6363 A service that supports Fragment-level access as described in 7.7 of this specification is encouraged  
6364 to support a subset of XPath at least as powerful as that described in Level 1.

6365 Clearly, the service can expose full XPath 1.0 or any other subset that meets or exceeds the  
6366 requirements defined here.

6367 A service that supports the Level 1 XPath dialect must ensure that it observes matching of a single  
6368 node. If more than one element of the same name is at the same level in the XML, the array notation  
6369 must be used to distinguish them.

### 6370 D.3 Level 2

6371 Level 2 contains everything defined in Level 1, plus general-purpose filtering functionality with the  
6372 standard set of relational operators and parenthesized sub-expressions (with AND, OR, NOT, and so  
6373 on). This dialect is suitable for filtering using enumerations and subscription filters. This dialect is a  
6374 strict superset of Level 1, with the <filter\_expression> production being considerably extended to  
6375 contain a useful subset of the XPath filtering syntax.

6376 EXAMPLE 1:

```
6377 (1) <path> ::= <root_selector> TOKEN_END_OF_INPUT;
6378 (2) <root_selector> ::= TOKEN_SLASH <element_sequence>;
6379 (3) <root_selector> ::= <relpath> <element_sequence>;
6380 (4) <root_selector> ::= <attribute>;
6381 (5) <root_selector> ::= TOKEN_DOT;
6382 (6) <relpath> ::= <> ;
6383 (7) <relpath> ::= TOKEN_DOT TOKEN_SLASH;
6384 (8) <relpath> ::= TOKEN_DOT_DOT TOKEN_SLASH;
6385 (9) <element_sequence> ::= <element> <optional_filter_expression> <more>;
6386 (10) <more> ::= TOKEN_SLASH <follower>;
6387 (11) <more> ::= <>;
6388 (12) <follower> ::= <attribute>;
6389 (13) <follower> ::= <text_function>;
6390 (14) <follower> ::= <element_sequence>;
6391 (15) <optional_filter_expression> ::= TOKEN_OPEN_BRACKET <filter_expression>
6392 TOKEN_CLOSE_BRACKET;
6393 (16) <optional_filter_expression> ::= <>;
6394 (17) <attribute> ::= TOKEN_AT_SYMBOL <name>;
6395 (18) <element> ::= <name>;
6396 (19) <text_function> ::= TOKEN_TEXT TOKEN_OPEN_PAREN TOKEN_CLOSE_PAREN;
6397 (20) <name> ::= TOKEN_XML_NAME;
6398 (21) <filter_expression> ::= <array_location>;
6399 (22) <array_location> ::= TOKEN_UNSIGNED_POSITIVE_INTEGER;
6400 (23) // Next level, simple OR expression
6401 (24) <or_expression> ::= <and_expression> <or_expression_rest>;
6402 (25) <or_expression_rest> ::= TOKEN_OR <and_expression> <or_expression_rest>;
6403 (26) <or_expression_rest> ::= <>;
6404 (27) // Next highest level, AND expression
6405 (28) <and_expression> ::= <rel_expression> <and_expression_rest>;
6406 (29) <and_expression_rest> ::= TOKEN_AND <rel_expression>
6407 <and_expression_rest>;
6408 (30) <and_expression_rest> ::= <>;
```

```

6409 (31) // Next level of precedence >, <, >=, <=, =, !=
6410 (32) <rel_expression> ::= <sub_expression> <rel_expression_rest>;
6411 (33) <rel_expression_rest> ::= <name> <rel_op> <const>;
6412 (34) <rel_expression_rest> ::= <>;
6413 (35) // Identifier, literal, or identifier + param_list (function call)
6414 (36) <sub_expression> ::= TOKEN_OPEN_PAREN <filter_expression>
6415 TOKEN_CLOSE_PAREN;
6416 (37) <sub_expression> ::= TOKEN_NOT TOKEN_OPEN_PAREN <filter_expression>
6417 TOKEN_CLOSE_PAREN;
6418 (38) // Relational operators
6419 (39) <rel_op> ::= TOKEN_GT; // >
6420 (40) <rel_op> ::= TOKEN_LT; // <
6421 (41) <rel_op> ::= TOKEN_GE; // >=
6422 (42) <rel_op> ::= TOKEN_LE; // <=
6423 (43) <rel_op> ::= TOKEN_EQ; // =
6424 (44) <rel_op> ::= TOKEN_NE; // !=
6425 (45) <const> ::= QUOTE TOKEN_STRING QUOTE;

```

6426 Terminals in the grammar are defined as shown in Table D-2.

6427

**Table D-2 – XPath Level 2 Terminals**

|                                 |                                         |
|---------------------------------|-----------------------------------------|
| TOKEN_SLASH                     | The character '/'                       |
| TOKEN_DOT                       | The character '.'                       |
| TOKEN_DOT_DOT                   | The characters '..'                     |
| TOKEN_END_OF_INPUT              | End of input                            |
| TOKEN_OPEN_BRACKET              | The character '['                       |
| TOKEN_CLOSE_BRACKET             | The character ']'                       |
| TOKEN_AT_SYMBOL                 | The character '@'                       |
| TOKEN_XML_NAME                  | Equivalent to XML Schema type xs:token  |
| TOKEN_UNSIGNED_POSITIVE_INTEGER | Values in the subrange 1..4294967295    |
| TOKEN_TEXT                      | The characters 'text'                   |
| TOKEN_OPEN_PAREN                | The character '('                       |
| TOKEN_CLOSE_PAREN               | The character ')'                       |
| TOKEN_AND                       | The characters 'and'                    |
| TOKEN_OR                        | The characters 'or'                     |
| TOKEN_NOT                       | The characters 'not'                    |
| TOKEN_STRING                    | Equivalent to XML Schema type xs:string |
| QUOTE                           | The character '"'                       |

6428 **EXAMPLE 2:** This dialect allows the same type of selection syntax as Level 1, but adds filtering, as in the  
6429 following generic examples, given the Level 1 example document above:

```

6430 (1) b[@x="y"] // Select if it has attribute x="y"
6431 (2) b[.="100"] // Select if it is 100
6432 (3) c[d="200"] // Select <c> if <d> is 200
6433 (4) c/d[.="200"] // Select <d> if it is 200
6434 (5) b[.="100" and @x="z"] // Select if it is 100 and has @x="z"
6435 (6) c[d="200" or d="300"] // Select all <c> with d=200 or d=300
6436 (7) c[2][not(.="400" or @x="100")]
6437 (8) // Select second <c> provided that:

```

```
6438 (9) // its value is not 400 and it does not have an attribute x set to 100
6439 (10) c/d[.="100" or (@x="400" and .="500")]
6440 (11) // Select <d> provided that:
6441 (12) // its value is 100 or it has an attribute x set to 400 and its value is
6442 500
```

6443 In essence, this dialect allows selecting any node based on a filter expression with the complete set  
6444 of relational operators, logical operators, and parenthesized sub-expressions.

6445 A service that supports XPath-based filtering dialects as described in this specification is encouraged  
6446 to support a subset of XPath at least as powerful as that described in Level 2.

6447 Clearly, the service can expose full XPath 1.0 or any other subset that meets or exceeds the  
6448 requirements defined here.

6449 In the actual operation, such as Enumerate or Subscribe, the XPath dialect is identified under the  
6450 normal URI for full XPath:

6451 <http://www.w3.org/TR/1999/REC-xpath-19991116>

6452  
6453  
6454  
6455

## ANNEX E (normative)

### Selector Filter Dialect

6456 The Selector filter dialect is a simple filtering dialect that allows a filtered enumeration or subscription  
6457 with no representation change.

6458 Selectors are part of the default addressing model as defined in 5.1. This dialect is intended for  
6459 implementations that support the default addressing model because it gives the ability to support  
6460 filtering using a similar syntax while avoiding additional processing overhead of supporting more  
6461 complex dialects.

6462 This specification defines the following dialect filter URI for the Selector dialect:

6463 `http://schemas.dmtf.org/wbem/wsman/1/wsman/SelectorFilter`

6464 If a service uses the WS-Management default addressing model, it can support this filter dialect for  
6465 enumeration and subscription operations.

6466 The Selector filter dialect can be used to specify name value pairs in the selector syntax to filter the  
6467 results from an Enumerate request or to identify the events of interest in a Subscribe request. The  
6468 selectors act as a selection mechanism against the resource class space implied by the  
6469 ResourceURI; however, there is no implication that the selector values are keys or even part of the  
6470 returned resource.

6471 The syntax for the filter in an Enumerate request is as follows:

```
6472 (1) <s:Header>
6473 (2) <wsa:To> Service transport address </wsa:To>
6474 (3) <wsman:ResourceURI> Resource URI </wsman:ResourceURI>
6475 (4) ...
6476 (5) </s:Header>
6477 (6) <s:Body>
6478 (7) <wsmen:Enumerate>
6479 (8) <wsman:Filter
6480 (9) Dialect="http://schemas.dmtf.org/wbem/wsman/1/wsman/SelectorFilter">
6481 (10) <wsman:SelectorSet>
6482 (11) <wsman:Selector Name="selector-name">
6483 (12) selector-value
6484 (13) </wsman:Selector> +
6485 (14) </wsman:SelectorSet>
6486 (15) </wsman:Filter>
6487 (16) ...
6488 (17) </wsmen:Enumerate>
6489 (18) </s:Body>
```

6490 Because the filter syntax does not include resource type information, the Resource URI specified in  
6491 the addressing block is used for identifying the resource type. Each of the individual selectors within a  
6492 SelectorSet are logically joined by AND for determining the result of the filter.

6493 **RE-1:** If the Selector Filter dialect is supported, a service shall accept as selector names the  
6494 local (NCName) part of the QNames of any of the top-level elements that represent the resource  
6495 instance or event and may accept additional selector names. If the service supports filtering only  
6496 on a subset of these QNames and the filter refers to an unsupported QName, the service shall  
6497 respond with a wsme:CannotProcessFilter fault (or wsman:CannotProcessFilter for Subscribe),

6498 and should provide in the fault detail the list of selector names that are supported for filtering by  
6499 the service.

6500 **RE-2:** For each selector name specified in the filter, the result of the operation shall contain  
6501 only instances for which that named element has the given value. Elements that are not  
6502 referenced from the filter can have any value.

6503 It is possible that some resource or event representations include elements of the same name, but  
6504 from different XML Namespaces. In this case, the service can choose to match on any of the  
6505 elements where the type matches the provided selector. Clients can be written to anticipate this, such  
6506 that there might be additional post-processing necessary to identify the set of desired instances.

6507 **RE-3:** If a resource or event representation includes two or more elements with QNames for  
6508 which the local part is identical but whose namespace names are different, and all of the following  
6509 conditions are present, the service shall not fault the request, and shall process the filter such that  
6510 it matches exactly one of the elements for which filtering is supported, using an algorithm of the  
6511 service's choosing:

- 6512 • A selector filter contains a wsman:Selector element whose Name attribute matches the  
6513 local part of each of these elements.
- 6514 • At least one of the matching elements has a type and value space consistent with the  
6515 provided selector type and value.
- 6516 • The service supports filtering on at least one of the corresponding elements per RE-1.

6517 **RE-4:** If a resource or event representation includes elements of an array type, and a filter  
6518 contains a wsman:Selector element whose Name attribute matches the local part of the QName  
6519 of these elements and the service supports filtering on the corresponding element per RE-1, the  
6520 service shall process the filter such that the results include all representations for which at least  
6521 one element of the array has a value equal to the value provided by the selector.

6522 Processing of the SelectorSet element when used as a filter follows the same processing rules as  
6523 when used in EPRs (as described in 5.4.2), with respect to duplicate selector names, type  
6524 mismatches, unexpected selectors, size restrictions, and so on.

6525 **RE-5:** If the filter expression contains a SelectorSet that is invalid with respect to the rules in  
6526 5.4.2, the service should fault with wsme:CannotProcessFilter (or wsman:CannotProcessFilter for  
6527 Subscribe) containing the appropriate detail code.



## ANNEX F (informative)

6528  
6529  
6530  
6531

### Identify XML Schema

6532 A normative copy of the XML schema of the Identify response message can be retrieved at the  
6533 following address:

6534 <http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd>

6535 The following non-normative copy of the XML schema is provided for convenience:

```
6536 (1) <?xml version="1.0" encoding="UTF-8"?>
6537 (2) <!--
6538 (3) Notice
6539 (4) DSP8012
6540 (5) Document: WS-Management Identify XML Schema
6541 (6) Version: 1.0.1
6542 (7) Status: Final
6543 (8) Date: 02/27/2009
6544 (9) Author: DMTF WS-Management Work Group Email:wsman-chair@dmtof.org
6545 (10) Description: XML Schema for WS-Management Identify Operation.
6546 (11)
6547 (12) Copyright © 2009 Distributed Management Task Force, Inc. (DMTF). All
6548 rights reserved. DMTF is a not-for-profit association of industry members
6549 dedicated to promoting enterprise and systems management and interoperability.
6550 Members and non-members may reproduce DMTF specifications and documents,
6551 provided that correct attribution is given. As DMTF specifications may be
6552 revised from time to time, the particular version and release date should
6553 always be noted. Implementation of certain elements of this standard or
6554 proposed standard may be subject to third party patent rights, including
6555 provisional patent rights (herein "patent rights"). DMTF makes no
6556 representations to users of the standard as to the existence of such rights,
6557 and is not responsible to recognize, disclose, or identify any or all such
6558 third party patent right, owners or claimants, nor for any incomplete or
6559 inaccurate identification or disclosure of such rights, owners or claimants.
6560 DMTF shall have no liability to any party, in any manner or circumstance, under
6561 any legal theory whatsoever, for failure to recognize, disclose, or identify
6562 any such third party patent rights, or for such party's reliance on the
6563 standard or incorporation thereof in its product, protocols or testing
6564 procedures. DMTF shall have no liability to any party implementing such
6565 standard, whether such implementation is foreseeable or not, nor to any patent
6566 owner or claimant, and shall have no liability or responsibility for costs or
6567 losses incurred if a standard is withdrawn or modified after publication, and
6568 shall be indemnified and held harmless by any party implementing the standard
6569 from any and all claims of infringement by a patent owner for such
6570 implementations. For information about patents held by third-parties which have
6571 notified the DMTF that, in their opinion, such patent may relate to or impact
6572 implementations of DMTF standards, visit
6573 http://www.dmtf.org/about/policies/disclosures.php.
6574 (13)
6575 (14) -->
6576 (15) <xs:schema
6577 (16) targetNamespace="http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanident
6578 ity.xsd"
6579 (17)
6580 xmlns:wsmid="http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd"
6581 (18) xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```

6582 (19) elementFormDefault="qualified" version="1.0.1">
6583 (20) <xs:complexType name="IdentifyType">
6584 (21) <xs:sequence>
6585 (22) <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
6586 (23) processContents="lax" />
6587 (24) </xs:sequence>
6588 (25) <xs:anyAttribute namespace="##other" processContents="lax" />
6589 (26) </xs:complexType>
6590 (27) <xs:element name="Identify" type="wsmid:IdentifyType" />
6591 (28)
6592 (29) <xs:simpleType name="restrictedProtocolVersionType">
6593 (30)
6594 (31) <xs:restriction base="xs:anyURI">
6595 (32) <xs:enumeration
6596 (33) value="http://schemas.dmtf.org/wbem/wsmid/identity/1/wsmid:IdentifyType/NoAnonymo
6597 usDisclosure" />
6598 (34) </xs:restriction>
6599 (35) </xs:simpleType>
6600 (36)
6601 (37) <xs:simpleType name="ProtocolVersionType">
6602 (38) <xs:union memberTypes="wsmid:restrictedProtocolVersionType xs:anyURI"
6603 />
6604 (39)
6605 (40) </xs:simpleType>
6606 (41) <xs:element name="ProtocolVersion" type="wsmid:ProtocolVersionType" />
6607 (42) <xs:element name="ProductVendor" type="xs:string" />
6608 (43) <xs:element name="ProductVersion" type="xs:string" />
6609 (44) <xs:element name="InitiativeName" type="xs:string" />
6610 (45) <xs:element name="InitiativeVersion" type="wsmid:VERSION_VALUE"/>
6611 (46) <xs:element name="SecurityProfileName" type="xs:anyURI" />
6612 (47) <xs:complexType name="SecurityProfilesType">
6613 (48) <xs:sequence>
6614 (49)
6615 (50) <xs:element ref="wsmid:SecurityProfileName" minOccurs="0"
6616 (51) maxOccurs="unbounded" />
6617 (52) </xs:sequence>
6618 (53) </xs:complexType>
6619 (54) <xs:element name="SecurityProfiles" type="wsmid:SecurityProfilesType" />
6620 (55) <xs:element name="AddressingVersionURI" type="xs:anyURI" />
6621 (56) <xs:element name="IntiativeSupport">
6622 (57) <xs:complexType>
6623 (58) <xs:sequence>
6624 (59) <xs:element ref="wsmid:InitiativeName" minOccurs="0" maxOccurs="1"
6625 (60) />
6626 (61) <xs:element ref="wsmid:InitiativeVersion" minOccurs="0"
6627 (62) maxOccurs="1"/>
6628 (63) </xs:sequence>
6629 (64) </xs:complexType>
6630 (65) </xs:element>
6631 (66) </xs:complexType name="IdentifyResponseType">
6632 (67) <xs:sequence>
6633 (68) <xs:element ref="wsmid:ProtocolVersion" maxOccurs="unbounded" />
6634 (69) <xs:element ref="wsmid:ProductVendor" minOccurs="0" />
6635 (70) <xs:element ref="wsmid:ProductVersion" minOccurs="0" />
6636 (71)

```

```
6640 (72) <xs:element ref="wsmid:InitiativeSupport" minOccurs="0"
6641 maxOccurs="unbounded"/>
6642 (73) <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" />
6643 (74) <xs:element ref="wsmid:SecurityProfiles" minOccurs="0"
6644 (75) maxOccurs="1" />
6645 (76) <xs:element ref="wsmid:AddressingVersionURI" minOccurs="0"
6646 (77) maxOccurs="unbounded" />
6647 (78) </xs:sequence>
6648 (79) <xs:anyAttribute namespace="##other" processContents="lax" />
6649 (80) </xs:complexType>
6650 (81)
6651 (82) <xs:element name="IdentifyResponse" type="wsmid:IdentifyResponseType" />
6652 (83)
6653 (84) <xs:simpleType name="VERSION_VALUE">
6654 (85)
6655 (86) <xs:annotation>
6656 (87) <xs:documentation>Version values must be in form of M.N.U (Major,
6657 Minor, Update)</xs:documentation>
6658 (88) </xs:annotation>
6659 (89) <xs:restriction base="xs:string">
6660 (90) <xs:pattern value="\d*.\d*.\d*" />
6661 (91) </xs:restriction>
6662 (92) </xs:simpleType>
6663 (93)
6664 (94) </xs:schema>
```

6665

## ANNEX G (informative)

6666  
6667  
6668  
6669

### Resource Access Operations XML Schema and WSDL

6670 A normative copy of the XML schemas (XML Schema 1, XML Schema 2) for the resource access  
6671 operations can be retrieved at the following address:

6672 [http://schemas.dmtf.org/wbem/wsman/1/DSP8031\\_1.0.xsd](http://schemas.dmtf.org/wbem/wsman/1/DSP8031_1.0.xsd)

6673 The following non-normative copy of the XML schema is provided for convenience:

```
6674 (1) <?xml version="1.0" encoding="UTF-8"?>
6675 (2) <!--
6676 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
6677 (4)
6678 (5) Document number: DSP8031
6679 (6) Date: 2010-02-19
6680 (7) Version: 1.0.0
6681 (8) Document status: DMTF Standard
6682 (9)
6683 (10) Title: WS-Management Resource Access Operations XML Schema
6684 (11)
6685 (12) Document type: Specification (W3C XML Schema)
6686 (13) Document language: E
6687 (14)
6688 (15) Abstract: XML Schema for WS-Management Resource Access Operations.
6689 (16)
6690 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
6691 (18)
6692 (19) Copyright (C) 2008-2010 Distributed Management Task Force, Inc. (DMTF).
6693 (20) All rights reserved. DMTF is a not-for-profit association of industry
6694 (21) members dedicated to promoting enterprise and systems management and
6695 (22) interoperability. Members and non-members may reproduce DMTF
6696 (23) specifications and documents, provided that correct attribution is
6697 (24) given. As DMTF specifications may be revised from time to time,
6698 (25) the particular version and release date should always be noted.
6699 (26) Implementation of certain elements of this standard or proposed
6700 (27) standard may be subject to third party patent rights, including
6701 (28) provisional patent rights (herein "patent rights"). DMTF makes
6702 (29) no representations to users of the standard as to the existence
6703 (30) of such rights, and is not responsible to recognize, disclose,
6704 (31) or identify any or all such third party patent right, owners or
6705 (32) claimants, nor for any incomplete or inaccurate identification or
6706 (33) disclosure of such rights, owners or claimants. DMTF shall have no
6707 (34) liability to any party, in any manner or circumstance, under any legal
6708 (35) theory whatsoever, for failure to recognize, disclose, or identify any
6709 (36) such third party patent rights, or for such party's reliance on the
6710 (37) standard or incorporation thereof in its product, protocols or testing
6711 (38) procedures. DMTF shall have no liability to any party implementing
6712 (39) such standard, whether such implementation is foreseeable or not, nor
6713 (40) to any patent owner or claimant, and shall have no liability or
6714 (41) responsibility for costs or losses incurred if a standard is withdrawn
6715 (42) or modified after publication, and shall be indemnified and held
6716 (43) harmless by any party implementing the standard from any and all claims
6717 (44) of infringement by a patent owner for such implementations. For
6718 (45) information about patents held by third-parties which have notified the
6719 (46) DMTF that, in their opinion, such patent may relate to or impact
```

```

6720 (47) implementations of DMTF standards, visit
6721 (48) http://www.dmtf.org/about/policies/disclosures.php.
6722 (49)
6723 (50) Change log:
6724 (51) 1.0.0 - 2009-11-01 - Work in Progress release
6725 (52) 1.0.0 - 2010-02-19 - DMTF Standard release
6726 (53)
6727 (54) -->
6728 (55) <xs:schema
6729 (56) targetNamespace="http://schemas.xmlsoap.org/ws/2004/09/transfer"
6730 (57) xmlns:tns="http://schemas.xmlsoap.org/ws/2004/09/transfer"
6731 (58) xmlns:xs="http://www.w3.org/2001/XMLSchema"
6732 (59) xmlns:wsa04="http://schemas.xmlsoap.org/ws/2004/08/addressing"
6733 (60) xmlns:wsa10="http://www.w3.org/2005/08/addressing"
6734 (61) elementFormDefault="qualified"
6735 (62) blockDefault="#all" >
6736 (63)
6737 (64) <xs:import
6738 (65) namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
6739 (66) schemaLocation="http://schemas.dmtf.org/wbem/wsman/1/DSP8034_1.0.xsd"
6740 (67) />
6741 (67) <xs:import
6742 (68) namespace="http://www.w3.org/2005/08/addressing"
6743 (69) schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd" />
6744 (70)
6745 (71) <!--
6746 (72) The type of the AnyEPRTType is effectively
6747 (73) the union of wsa04:EndpointReferenceType and
6748 (74) wsa10:EndpointReferenceType. Unfortunately, xs:union only
6749 (75) works for simple types. As a result, we have to define
6750 (76) the element in an unvalidated way to accommodate either
6751 (77) addressing type.
6752 (78) -->
6753 (79)
6754 (80) <xs:complexType name="AnyEPRTType">
6755 (81) <xs:sequence>
6756 (82) <xs:any minOccurs='1' maxOccurs='unbounded' processContents='skip'
6757 (83) namespace='##other' />
6758 (84) </xs:sequence>
6759 (85) </xs:complexType>
6760 (86)
6761 (87) <xs:element name="ResourceCreated" type="tns:AnyEPRTType"/>
6762 (88)
6763 (89) <!-- The following GED is defined for convenience. This GED
6764 (90) may be used in cases where a resource-specific GED is
6765 (91) not available. -->
6766 (92) <xs:element name="TransferElement">
6767 (93) <xs:complexType>
6768 (94) <xs:sequence>
6769 (95) <xs:any minOccurs='1' maxOccurs='unbounded'
6770 (96) processContents='skip' namespace='##other' />
6771 (97) </xs:sequence>
6772 (98) </xs:complexType>
6773 (99) </xs:element>
6774 (100)
6775 (101) </xs:schema>

```

6776 A normative copy of the WSDL description for the resource access operations can be retrieved from  
6777 the following address:

6778 [http://schemas.dmtf.org/wbem/wsman/1/DSP8035\\_1.0.wsdl](http://schemas.dmtf.org/wbem/wsman/1/DSP8035_1.0.wsdl)

6779 The following non-normative copy of the WSDL description is provided for convenience:

```
6780 (1) <?xml version="1.0" encoding="UTF-8"?>
6781 (2) <!--
6782 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
6783 (4)
6784 (5) Document number: DSP8035
6785 (6) Date: 2010-02-19
6786 (7) Version: 1.0.0
6787 (8) Document status: DMTF Standard
6788 (9)
6789 (10) Title: WS-Management Resource Access Operations WSDL
6790 (11)
6791 (12) Document type: Specification (W3C WSDL Document)
6792 (13) Document language: E
6793 (14)
6794 (15) Abstract: WSDL for WS-Management Resource Access Operations.
6795 (16)
6796 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
6797 (18)
6798 (19) Copyright (C) 2008-2010 Distributed Management Task Force, Inc. (DMTF).
6799 (20) All rights reserved. DMTF is a not-for-profit association of industry
6800 (21) members dedicated to promoting enterprise and systems management and
6801 (22) interoperability. Members and non-members may reproduce DMTF
6802 (23) specifications and documents, provided that correct attribution is
6803 (24) given. As DMTF specifications may be revised from time to time,
6804 (25) the particular version and release date should always be noted.
6805 (26) Implementation of certain elements of this standard or proposed
6806 (27) standard may be subject to third party patent rights, including
6807 (28) provisional patent rights (herein "patent rights"). DMTF makes
6808 (29) no representations to users of the standard as to the existence
6809 (30) of such rights, and is not responsible to recognize, disclose,
6810 (31) or identify any or all such third party patent right, owners or
6811 (32) claimants, nor for any incomplete or inaccurate identification or
6812 (33) disclosure of such rights, owners or claimants. DMTF shall have no
6813 (34) liability to any party, in any manner or circumstance, under any legal
6814 (35) theory whatsoever, for failure to recognize, disclose, or identify any
6815 (36) such third party patent rights, or for such party's reliance on the
6816 (37) standard or incorporation thereof in its product, protocols or testing
6817 (38) procedures. DMTF shall have no liability to any party implementing
6818 (39) such standard, whether such implementation is foreseeable or not, nor
6819 (40) to any patent owner or claimant, and shall have no liability or
6820 (41) responsibility for costs or losses incurred if a standard is withdrawn
6821 (42) or modified after publication, and shall be indemnified and held
6822 (43) harmless by any party implementing the standard from any and all claims
6823 (44) of infringement by a patent owner for such implementations. For
6824 (45) information about patents held by third-parties which have notified the
6825 (46) DMTF that, in their opinion, such patent may relate to or impact
6826 (47) implementations of DMTF standards, visit
6827 (48) http://www.dmtf.org/about/policies/disclosures.php.
6828 (49)
6829 (50) Change log:
6830 (51) 1.0.0 - 2009-11-01 - Work in Progress release
6831 (52) 1.0.0 - 2010-02-19 - DMTF Standard release
6832 (53)
6833 (54) -->
6834 (55) <wsdl:definitions
6835 (56) targetNamespace="http://schemas.xmlsoap.org/ws/2004/09/transfer"
```

```

6836 (57) xmlns:tns="http://schemas.xmlsoap.org/ws/2004/09/transfer"
6837 (58) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
6838 (59) xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
6839 (60) xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
6840 (61) xmlns:xs="http://www.w3.org/2001/XMLSchema">
6841 (62)
6842 (63) <wSDL:types>
6843 (64) <xs:schema>
6844 (65) <xs:import
6845 (66) namespace="http://schemas.xmlsoap.org/ws/2004/09/transfer"
6846 (67)
6847 schemaLocation="http://schemas.dmtf.org/wbem/wsman/1/DSP8031_1.0.xsd"
6848 (68) />
6849 (69) </xs:schema>
6850 (70) </wSDL:types>
6851 (71)
6852 (72) <!--
6853 (73) In some of the messages defined below a "resource-specific-GED"
6854 (74) is expected to be inserted before the WSDL is processed by any tooling.
6855 (75) Thus the WSDL as presented is not usable until after this substitution
6856 (76) is done.
6857 (77) -->
6858 (78)
6859 (79) <wSDL:message name="EmptyMessage"/>
6860 (80) <wSDL:message name="CreateRequestMessage">
6861 (81) <wSDL:part name="Body" element="resource-specific-GED"/>
6862 (82) </wSDL:message>
6863 (83) <wSDL:message name="CreateResponseMessage">
6864 (84) <wSDL:part name="Body" element="tns:ResourceCreated"/>
6865 (85) </wSDL:message>
6866 (86) <wSDL:message name="GetResponseMessage">
6867 (87) <wSDL:part name="Body" element="resource-specific-GED"/>
6868 (88) </wSDL:message>
6869 (89) <wSDL:message name="PutRequestMessage">
6870 (90) <wSDL:part name="Body" element="resource-specific-GED"/>
6871 (91) </wSDL:message>
6872 (92) <wSDL:message name="PutResponseMessage">
6873 (93) <!-- Note this 'part' may be omitted -->
6874 (94) <wSDL:part name="Body" element="resource-specific-GED"/>
6875 (95) </wSDL:message>
6876 (96)
6877 (97) <wSDL:portType name="Resource">
6878 (98) <wSDL:documentation>
6879 (99) This port type defines a resource that may be read,
6880 (100) written, and deleted.
6881 (101) </wSDL:documentation>
6882 (102) <wSDL:operation name="Get">
6883 (103) <wSDL:input
6884 (104) message="tns:EmptyMessage"
6885 (105) wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Get"
6886 (106) wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Get"
6887 />
6888 (107) <wSDL:output
6889 (108) message="tns:GetResponseMessage"
6890 (109)
6891 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse"
6892 (110)
6893 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse" />
6894 (111) </wSDL:operation>
6895 (112) <wSDL:operation name="Put">
6896 (113) <wSDL:input
6897 (114) message="tns:PutRequestMessage"
6898 (115) wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Put"

```

```
6899 (116) wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Put"
6900 />
6901 (117) <wsdl:output
6902 (118) message="tns:PutResponseMessage"
6903 (119)
6904 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/PutResponse"
6905 (120)
6906 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/PutResponse"/>
6907 (121) </wsdl:operation>
6908 (122) <wsdl:operation name="Delete">
6909 (123) <wsdl:input
6910 (124) message="tns:EmptyMessage"
6911 (125)
6912 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Delete"
6913 (126)
6914 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Delete" />
6915 (127) <wsdl:output
6916 (128) message="tns:EmptyMessage"
6917 (129)
6918 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/DeleteResponse"
6919 (130)
6920 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/DeleteResponse"
6921 (131) />
6922 (132) </wsdl:operation>
6923 (133) </wsdl:portType>
6924 (134)
6925 (135) <wsdl:portType name="ResourceFactory">
6926 (136) <wsdl:documentation>
6927 (137) This port type defines a Web service that can create new
6928 (138) resources.
6929 (139) </wsdl:documentation>
6930 (140) <wsdl:operation name="Create">
6931 (141) <wsdl:input
6932 (142) message="tns:CreateRequestMessage"
6933 (143)
6934 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Create"
6935 (144)
6936 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/Create" />
6937 (145) <wsdl:output
6938 (146) message="tns:CreateResponseMessage"
6939 (147)
6940 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse"
6941 (148)
6942 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/transfer/CreateResponse"
6943 (149) />
6944 (150) </wsdl:operation>
6945 (151) </wsdl:portType>
6946 (152) </wsdl:definitions>
```



## ANNEX H (informative)

6947  
6948  
6949  
6950

### Enumeration Operations XML Schema and WSDL

6951 A normative copy of the XML schemas for the enumeration operations can be retrieved at the  
6952 following address:

6953 [http://schemas.dmtf.org/wbem/wsman/1/DSP8033\\_1.0.xsd](http://schemas.dmtf.org/wbem/wsman/1/DSP8033_1.0.xsd)

6954 The following non-normative copy of the XML schema is provided for convenience:

```
6955 (1) <?xml version="1.0" encoding="UTF-8"?>
6956 (2) <!--
6957 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
6958 (4)
6959 (5) Document number: DSP8033
6960 (6) Date: 2010-02-19
6961 (7) Version: 1.0.0
6962 (8) Document status: DMTF Standard
6963 (9)
6964 (10) Title: WS-Management Enumeration Operations XML Schema
6965 (11)
6966 (12) Document type: Specification (W3C XML Schema)
6967 (13) Document language: E
6968 (14)
6969 (15) Abstract: XML Schema for WS-Management Enumeration Operations.
6970 (16)
6971 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
6972 (18)
6973 (19) Copyright (C) 2008-2010 Distributed Management Task Force, Inc. (DMTF).
6974 (20) All rights reserved. DMTF is a not-for-profit association of industry
6975 (21) members dedicated to promoting enterprise and systems management and
6976 (22) interoperability. Members and non-members may reproduce DMTF
6977 (23) specifications and documents, provided that correct attribution is
6978 (24) given. As DMTF specifications may be revised from time to time,
6979 (25) the particular version and release date should always be noted.
6980 (26) Implementation of certain elements of this standard or proposed
6981 (27) standard may be subject to third party patent rights, including
6982 (28) provisional patent rights (herein "patent rights"). DMTF makes
6983 (29) no representations to users of the standard as to the existence of
6984 (30) such rights, and is not responsible to recognize, disclose,
6985 (31) or identify any or all such third party patent right, owners or
6986 (32) claimants, nor for any incomplete or inaccurate identification or
6987 (33) disclosure of such rights, owners or claimants. DMTF shall have no
6988 (34) liability to any party, in any manner or circumstance, under any legal
6989 (35) theory whatsoever, for failure to recognize, disclose, or identify any
6990 (36) such third party patent rights, or for such party's reliance on the
6991 (37) standard or incorporation thereof in its product, protocols or testing
6992 (38) procedures. DMTF shall have no liability to any party implementing
6993 (39) such standard, whether such implementation is foreseeable or not, nor
6994 (40) to any patent owner or claimant, and shall have no liability or
6995 (41) responsibility for costs or losses incurred if a standard is withdrawn
6996 (42) or modified after publication, and shall be indemnified and held
6997 (43) harmless by any party implementing the standard from any and all claims
6998 (44) of infringement by a patent owner for such implementations. For
6999 (45) information about patents held by third-parties which have notified the
7000 (46) DMTF that, in their opinion, such patent may relate to or impact
7001 (47) implementations of DMTF standards, visit
7002 (48) http://www.dmtf.org/about/policies/disclosures.php.
```

```
7003 (49)
7004 (50) Change log:
7005 (51) 1.0.0 - 2009-11-01 - Work in Progress release
7006 (52) 1.0.0 - 2010-02-19 - DMTF Standard release
7007 (53)
7008 (54) -->
7009 (55) <xs:schema
7010 (56) targetNamespace="http://schemas.xmlsoap.org/ws/2004/09/enumeration"
7011 (57) xmlns:tns="http://schemas.xmlsoap.org/ws/2004/09/enumeration"
7012 (58) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
7013 (59) xmlns:xs="http://www.w3.org/2001/XMLSchema"
7014 (60) elementFormDefault="qualified"
7015 (61) blockDefault="#all">
7016 (62)
7017 (63) <xs:import
7018 (64) namespace="http://www.w3.org/XML/1998/namespace"
7019 (65) schemaLocation="http://www.w3.org/2001/xml.xsd" />
7020 (66) <xs:import
7021 (67) namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
7022 (68) schemaLocation="http://schemas.dmtf.org/wbem/wsman/1/DSP8034_1.0.xsd"
7023 />
7024 (69) <xs:import
7025 (70) namespace="http://www.w3.org/2005/08/addressing"
7026 (71) schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd" />
7027 (72)
7028 (73) <!-- Types and global elements -->
7029 (74) <xs:complexType name="FilterType" mixed="true">
7030 (75) <xs:sequence>
7031 (76) <xs:any namespace="##other" processContents="lax"
7032 (77) minOccurs="0" maxOccurs="unbounded" />
7033 (78) </xs:sequence>
7034 (79) <xs:attribute name="Dialect" type="xs:anyURI" />
7035 (80) <xs:anyAttribute namespace="##other" processContents="lax" />
7036 (81) </xs:complexType>
7037 (82)
7038 (83) <xs:simpleType name="PositiveDurationType">
7039 (84) <xs:restriction base="xs:duration">
7040 (85) <xs:minExclusive value="P0Y0M0DT0H0M0S" />
7041 (86) </xs:restriction>
7042 (87) </xs:simpleType>
7043 (88)
7044 (89) <xs:simpleType name="NonNegativeDurationType">
7045 (90) <xs:restriction base="xs:duration">
7046 (91) <xs:minInclusive value="P0Y0M0DT0H0M0S" />
7047 (92) </xs:restriction>
7048 (93) </xs:simpleType>
7049 (94)
7050 (95) <xs:simpleType name="ExpirationType">
7051 (96) <xs:union memberTypes="xs:dateTime tns:NonNegativeDurationType" />
7052 (97) </xs:simpleType>
7053 (98)
7054 (99) <xs:complexType name="EnumerationContextType">
7055 (100) <xs:complexContent mixed="true">
7056 (101) <xs:restriction base="xs:anyType">
7057 (102) <xs:sequence>
7058 (103) <xs:any namespace="##other" processContents="lax"
7059 (104) minOccurs="0" maxOccurs="unbounded" />
7060 (105) </xs:sequence>
7061 (106) <xs:anyAttribute namespace="##other" processContents="lax" />
7062 (107) </xs:restriction>
7063 (108) </xs:complexContent>
7064 (109) </xs:complexType>
7065 (110)
```

```

7066 (111) <xs:complexType name="ItemListType">
7067 (112) <xs:sequence maxOccurs="unbounded">
7068 (113) <xs:any namespace="##other" processContents="lax"
7069 (114) minOccurs="0" maxOccurs="unbounded" />
7070 (115) </xs:sequence>
7071 (116) </xs:complexType>
7072 (117)
7073 (118) <xs:complexType name="LanguageSpecificStringType">
7074 (119) <xs:simpleContent>
7075 (120) <xs:extension base="xs:string">
7076 (121) <xs:attribute ref="xml:lang" />
7077 (122) <xs:anyAttribute namespace="##other" processContents="lax" />
7078 (123) </xs:extension>
7079 (124) </xs:simpleContent>
7080 (125) </xs:complexType>
7081 (126)
7082 (127) <!--
7083 (128) The type of the AnyEPRType is effectively
7084 (129) the union of wsa04:EndpointReferenceType and
7085 (130) wsa10:EndpointReferenceType. Unfortunately, xs:union only
7086 (131) works for simple types. As a result, we have to define
7087 (132) the element in an unvalidated way to accommodate either
7088 (133) addressing type.
7089 (134) -->
7090 (135)
7091 (136) <xs:complexType name="AnyEPRType">
7092 (137) <xs:sequence>
7093 (138) <xs:any minOccurs='1' maxOccurs='unbounded' processContents='skip'
7094 (139) namespace='##other' />
7095 (140) </xs:sequence>
7096 (141) </xs:complexType>
7097 (142)
7098 (143) <!-- Enumerate request -->
7099 (144) <xs:element name="Enumerate">
7100 (145) <xs:complexType>
7101 (146) <xs:sequence>
7102 (147) <xs:element name="EndTo" type="tns:AnyEPRType"
7103 (148) minOccurs="0" />
7104 (149) <xs:element name="Expires" type="tns:ExpirationType"
7105 (150) minOccurs="0" />
7106 (151) <xs:element name="Filter" type="tns:FilterType"
7107 (152) minOccurs="0" />
7108 (153) <xs:any namespace="##other" processContents="lax"
7109 (154) minOccurs="0" maxOccurs="unbounded" />
7110 (155) </xs:sequence>
7111 (156) <xs:anyAttribute namespace="##other" processContents="lax" />
7112 (157) </xs:complexType>
7113 (158) </xs:element>
7114 (159)
7115 (160) <!-- Used for a fault response -->
7116 (161) <xs:element name="SupportedDialect" type="xs:anyURI" />
7117 (162)
7118 (163) <!-- Enumerate response -->
7119 (164) <xs:element name="EnumerateResponse">
7120 (165) <xs:complexType>
7121 (166) <xs:sequence>
7122 (167) <xs:element name="Expires" type="tns:ExpirationType"
7123 (168) minOccurs="0" />
7124 (169) <xs:element name="EnumerationContext"
7125 (170) type="tns:EnumerationContextType" />
7126 (171) <xs:any namespace="##other" processContents="lax"
7127 (172) minOccurs="0" maxOccurs="unbounded" />
7128 (173) </xs:sequence>

```

```

7129 (174) <xs:anyAttribute namespace="##other" processContents="lax" />
7130 (175) </xs:complexType>
7131 (176) </xs:element>
7132 (177)
7133 (178) <!-- Pull request -->
7134 (179) <xs:element name="Pull">
7135 (180) <xs:complexType>
7136 (181) <xs:sequence>
7137 (182) <xs:element name="EnumerationContext"
7138 (183) type="tns:EnumerationContextType" />
7139 (184) <xs:element name="MaxTime" type="tns:PositiveDurationType"
7140 (185) minOccurs="0" />
7141 (186) <xs:element name="MaxElements" type="xs:positiveInteger"
7142 (187) minOccurs="0" />
7143 (188) <xs:element name="MaxCharacters" type="xs:positiveInteger"
7144 (189) minOccurs="0" />
7145 (190) <xs:any namespace="##other" processContents="lax"
7146 (191) minOccurs="0" maxOccurs="unbounded" />
7147 (192) </xs:sequence>
7148 (193) <xs:anyAttribute namespace="##other" processContents="lax" />
7149 (194) </xs:complexType>
7150 (195) </xs:element>
7151 (196)
7152 (197) <!-- Pull response -->
7153 (198) <xs:element name="PullResponse">
7154 (199) <xs:complexType>
7155 (200) <xs:sequence>
7156 (201) <xs:element name="EnumerationContext"
7157 (202) type="tns:EnumerationContextType"
7158 (203) minOccurs="0" />
7159 (204) <xs:element name="Items" type="tns:ItemListType"
7160 (205) minOccurs="0" />
7161 (206) <xs:element name="EndOfSequence" minOccurs="0" />
7162 (207) </xs:sequence>
7163 (208) <xs:anyAttribute namespace="##other" processContents="lax" />
7164 (209) </xs:complexType>
7165 (210) </xs:element>
7166 (211)
7167 (212) <!-- Renew request -->
7168 (213) <xs:element name="Renew">
7169 (214) <xs:complexType>
7170 (215) <xs:sequence>
7171 (216) <xs:element name="EnumerationContext"
7172 (217) type="tns:EnumerationContextType" />
7173 (218) <xs:element name="Expires" type="tns:ExpirationType"
7174 (219) minOccurs="0" />
7175 (220) <xs:any namespace="##other" processContents="lax"
7176 (221) minOccurs="0" maxOccurs="unbounded" />
7177 (222) </xs:sequence>
7178 (223) <xs:anyAttribute namespace="##other" processContents="lax" />
7179 (224) </xs:complexType>
7180 (225) </xs:element>
7181 (226)
7182 (227) <!-- Renew response -->
7183 (228) <xs:element name="RenewResponse">
7184 (229) <xs:complexType>
7185 (230) <xs:sequence>
7186 (231) <xs:element name="Expires" type="tns:ExpirationType"
7187 (232) minOccurs="0" />
7188 (233) <xs:element name="EnumerationContext"
7189 (234) type="tns:EnumerationContextType"
7190 (235) minOccurs="0" />
7191 (236) <xs:any namespace="##other" processContents="lax"

```

```

7192 minOccurs="0" maxOccurs="unbounded" />
7193 </xs:sequence>
7194 <xs:anyAttribute namespace="##other" processContents="lax" />
7195 </xs:complexType>
7196 </xs:element>
7197 (242)
7198 <!-- GetStatus request -->
7199 <xs:element name="GetStatus">
7200 <xs:complexType>
7201 <xs:sequence>
7202 <xs:element name="EnumerationContext"
7203 type="tns:EnumerationContextType" />
7204 <xs:any namespace="##other" processContents="lax"
7205 minOccurs="0" maxOccurs="unbounded" />
7206 </xs:sequence>
7207 <xs:anyAttribute namespace="##other" processContents="lax" />
7208 </xs:complexType>
7209 </xs:element>
7210 (255)
7211 <!-- GetStatus response -->
7212 <xs:element name="GetStatusResponse">
7213 <xs:complexType>
7214 <xs:sequence>
7215 <xs:element name="Expires" type="tns:ExpirationType"
7216 minOccurs="0" />
7217 <xs:any namespace="##other" processContents="lax"
7218 minOccurs="0" maxOccurs="unbounded" />
7219 </xs:sequence>
7220 <xs:anyAttribute namespace="##other" processContents="lax" />
7221 </xs:complexType>
7222 </xs:element>
7223 (268)
7224 <!-- Release request -->
7225 <xs:element name="Release">
7226 <xs:complexType>
7227 <xs:sequence>
7228 <xs:element name="EnumerationContext"
7229 type="tns:EnumerationContextType" />
7230 </xs:sequence>
7231 <xs:anyAttribute namespace="##other" processContents="lax" />
7232 </xs:complexType>
7233 </xs:element>
7234 (279)
7235 <!-- Release response has an empty body -->
7236 (281)
7237 <!-- EnumerationEnd message -->
7238 <xs:element name="EnumerationEnd">
7239 <xs:complexType>
7240 <xs:sequence>
7241 <xs:element name="EnumerationContext"
7242 type="tns:EnumerationContextType" />
7243 <xs:element name="Code" type="tns:OpenEnumerationEndCodeType" />
7244 <xs:element name="Reason" type="tns:LanguageSpecificStringType"
7245 minOccurs="0" maxOccurs="unbounded" />
7246 <xs:any namespace="##other" processContents="lax"
7247 minOccurs="0" maxOccurs="unbounded" />
7248 </xs:sequence>
7249 <xs:anyAttribute namespace="##other" processContents="lax" />
7250 </xs:complexType>
7251 </xs:element>
7252 (297)
7253 <xs:simpleType name="EnumerationEndCodeType">
7254 <xs:restriction base="xs:anyURI">

```

```

7255 (300) <xs:enumeration
7256 value="http://schemas.xmlsoap.org/ws/2004/09/enumeration/SourceShuttingDown" />
7257 (301) <xs:enumeration
7258 value="http://schemas.xmlsoap.org/ws/2004/09/enumeration/SourceCancelling" />
7259 (302) </xs:restriction>
7260 (303) </xs:simpleType>
7261 (304)
7262 (305) <xs:simpleType name="OpenEnumerationEndCodeType">
7263 (306) <xs:union memberTypes="tns:EnumerationEndCodeType xs:anyURI" />
7264 (307) </xs:simpleType>
7265 (308) </xs:schema>

```

7266 A normative copy of the WSDL description for enumeration operations can be retrieved from the  
7267 following address:

7268 [http://schemas.dmtf.org/wbem/wsman/1/DSP8037\\_1.0.wsdl](http://schemas.dmtf.org/wbem/wsman/1/DSP8037_1.0.wsdl)

7269 The following non-normative copy of the WSDL description is provided for convenience:

```

7270 (1) <?xml version="1.0" encoding="UTF-8"?>
7271 (2) <!--
7272 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
7273 (4)
7274 (5) Document number: DSP8037
7275 (6) Date: 2010-02-19
7276 (7) Version: 1.0.0
7277 (8) Document status: DMTF Standard
7278 (9)
7279 (10) Title: WS-Management Enumeration Operations WSDL
7280 (11)
7281 (12) Document type: Specification (W3C WSDL Document)
7282 (13) Document language: E
7283 (14)
7284 (15) Abstract: WSDL for WS-Management Enumeration Operations.
7285 (16)
7286 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
7287 (18)
7288 (19) Copyright (C) 2008-2010 Distributed Management Task Force, Inc. (DMTF).
7289 (20) All rights reserved. DMTF is a not-for-profit association of industry
7290 (21) members dedicated to promoting enterprise and systems management and
7291 (22) interoperability. Members and non-members may reproduce DMTF
7292 (23) specifications and documents, provided that correct attribution is
7293 (24) given. As DMTF specifications may be revised from time to time,
7294 (25) the particular version and release date should always be noted.
7295 (26) Implementation of certain elements of this standard or proposed
7296 (27) standard may be subject to third party patent rights, including
7297 (28) provisional patent rights (herein "patent rights"). DMTF makes
7298 (29) no representations to users of the standard as to the existence of
7299 (30) such rights, and is not responsible to recognize, disclose,
7300 (31) or identify any or all such third party patent right, owners or
7301 (32) claimants, nor for any incomplete or inaccurate identification or
7302 (33) disclosure of such rights, owners or claimants. DMTF shall have no
7303 (34) liability to any party, in any manner or circumstance, under any legal
7304 (35) theory whatsoever, for failure to recognize, disclose, or identify any
7305 (36) such third party patent rights, or for such party's reliance on the
7306 (37) standard or incorporation thereof in its product, protocols or testing
7307 (38) procedures. DMTF shall have no liability to any party implementing
7308 (39) such standard, whether such implementation is foreseeable or not, nor
7309 (40) to any patent owner or claimant, and shall have no liability or
7310 (41) responsibility for costs or losses incurred if a standard is withdrawn
7311 (42) or modified after publication, and shall be indemnified and held
7312 (43) harmless by any party implementing the standard from any and all claims
7313 (44) of infringement by a patent owner for such implementations. For

```

```
7314 (45) information about patents held by third-parties which have notified the
7315 (46) DMTF that, in their opinion, such patent may relate to or impact
7316 (47) implementations of DMTF standards, visit
7317 (48) http://www.dmtf.org/about/policies/disclosures.php.
7318 (49)
7319 (50) Change log:
7320 (51) 1.0.0 - 2009-11-01 - Work in Progress release
7321 (52) 1.0.0 - 2010-02-19 - DMTF Standard release
7322 (53)
7323 (54) -->
7324 (55) <wsdl:definitions
7325 (56) targetNamespace="http://schemas.xmlsoap.org/ws/2004/09/enumeration"
7326 (57) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
7327 (58) xmlns:wsm="http://www.w3.org/2007/05/addressing/metadata"
7328 (59) xmlns:wsmen="http://schemas.xmlsoap.org/ws/2004/09/enumeration"
7329 (60) xmlns:wsmen="http://schemas.xmlsoap.org/ws/2004/09/enumeration"
7330 (61) xmlns:xs="http://www.w3.org/2001/XMLSchema" >
7331 (62)
7332 (63) <wsdl:types>
7333 (64) <xs:schema
7334 (65) <xs:import
7335 (66) namespace="http://schemas.xmlsoap.org/ws/2004/09/enumeration"
7336 (67) schemaLocation="http://schemas.dmtf.org/wbem/wsman/1/DSP8033_1.0.xsd"
7337 (68) />
7338 (69) </xs:schema>
7339 (70) </wsdl:types>
7340 (71)
7341 (72) <wsdl:message name="EnumerateMessage">
7342 (73) <wsdl:part name="Body" element="wsmen:Enumerate" />
7343 (74) </wsdl:message>
7344 (75) <wsdl:message name="EnumerateResponseMessage">
7345 (76) <wsdl:part name="Body" element="wsmen:EnumerateResponse" />
7346 (77) </wsdl:message>
7347 (78) <wsdl:message name="PullMessage">
7348 (79) <wsdl:part name="Body" element="wsmen:Pull" />
7349 (80) </wsdl:message>
7350 (81) <wsdl:message name="PullResponseMessage">
7351 (82) <wsdl:part name="Body" element="wsmen:PullResponse" />
7352 (83) </wsdl:message>
7353 (84) <wsdl:message name="RenewMessage" >
7354 (85) <wsdl:part name="Body" element="wsmen:Renew" />
7355 (86) </wsdl:message>
7356 (87) <wsdl:message name="RenewResponseMessage" >
7357 (88) <wsdl:part name="Body" element="wsmen:RenewResponse" />
7358 (89) </wsdl:message>
7359 (90) <wsdl:message name="GetStatusMessage" >
7360 (91) <wsdl:part name="Body" element="wsmen:GetStatus" />
7361 (92) </wsdl:message>
7362 (93) <wsdl:message name="GetStatusResponseMessage" >
7363 (94) <wsdl:part name="Body" element="wsmen:GetStatusResponse" />
7364 (95) </wsdl:message>
7365 (96) <wsdl:message name="ReleaseMessage">
7366 (97) <wsdl:part name="Body" element="wsmen:Release" />
7367 (98) </wsdl:message>
7368 (99) <wsdl:message name="ReleaseResponseMessage" />
7369 (100) <wsdl:message name="EnumerationEndMessage" >
7370 (101) <wsdl:part name="Body" element="wsmen:EnumerationEnd" />
7371 (102) </wsdl:message>
7372 (103)
7373 (104) <wsdl:portType name="DataSource">
7374 (105) <wsdl:operation name="EnumerateOp">
7375 (106) <wsdl:input
7376 (107) message="wsmen:EnumerateMessage"
```

```

7377 (108)
7378 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate"
7379 (109)
7380 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate"
7381 (110) />
7382 (111) <wsdl:output
7383 (112) message="wsmen:EnumerateResponseMessage"
7384 (113)
7385 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerateResponse"
7386 "
7387 (114)
7388 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerateResponse"
7389 "
7390 (115) />
7391 (116) </wsdl:operation>
7392 (117) <wsdl:operation name="PullOp">
7393 (118) <wsdl:input
7394 (119) message="wsmen:PullMessage"
7395 (120)
7396 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Pull"
7397 (121)
7398 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Pull"
7399 (122) />
7400 (123) <wsdl:output
7401 (124) message="wsmen:PullResponseMessage"
7402 (125)
7403 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/PullResponse"
7404 (126)
7405 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/PullResponse"
7406 (127) />
7407 (128) </wsdl:operation>
7408 (129) <wsdl:operation name="RenewOp" >
7409 (130) <wsdl:input
7410 (131) message="wsmen:RenewMessage"
7411 (132)
7412 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Renew"
7413 (133)
7414 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Renew"
7415 (134) />
7416 (135) <wsdl:output
7417 (136) message="wsmen:RenewResponseMessage"
7418 (137)
7419 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/RenewResponse"
7420 (138) wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/RenewResponse"
7421 "
7422 (139) />
7423 (140) </wsdl:operation>
7424 (141) <wsdl:operation name="GetStatusOp" >
7425 (142) <wsdl:input
7426 (143) message="wsmen:GetStatusMessage"
7427 (144)
7428 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatus"
7429 (145)
7430 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatus"
7431 (146) />
7432 (147) <wsdl:output
7433 (148) message="wsmen:GetStatusResponseMessage"
7434 (149)
7435 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatusResponse"
7436 "
7437 (150) wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/GetStatusResponse"
7438 "
7439 (151) />

```



```
7440 (152) </wsdl:operation>
7441 (153) <wsdl:operation name="ReleaseOp">
7442 (154) <wsdl:input
7443 (155) message="wsmen:ReleaseMessage"
7444 (156)
7445 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Release"
7446 (157)
7447 wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Release"
7448 (158) />
7449 (159) <wsdl:output
7450 (160) message="wsmen:ReleaseResponseMessage"
7451 (161) wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/ReleaseResp
7452 onse"
7453 (162) wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/ReleaseRes
7454 ponse"
7455 (163) />
7456 (164) </wsdl:operation>
7457 (165) </wsdl:portType>
7458 (166)
7459 (167) <!-- The following portType shall be supported by the endpoint to which
7460 (168) The EnumerationEnd message is sent -->
7461 (169) <wsdl:portType name="EnumEndEndpoint">
7462 (170) <wsdl:operation name="EnumerationEndOp" >
7463 (171) <wsdl:input
7464 (172) message="wsmen:EnumerationEndMessage"
7465 (173)
7466 wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/EnumerationEnd"
7467 (174) wsam:Action="http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumeratio
7468 nEnd"
7469 (175) />
7470 (176) </wsdl:operation>
7471 (177) </wsdl:portType>
7472 (178) </wsdl:definitions>
```

7473

## ANNEX I (informative)

7474  
7475  
7476  
7477

### Notification OperationsXML Schema and WSDL

7478 A normative copy of the XML schemas for the notification operations can be retrieved at the following  
7479 address:

7480 [http://schemas.dmtf.org/wbem/wsman/1/DSP8032\\_1.0.xsd](http://schemas.dmtf.org/wbem/wsman/1/DSP8032_1.0.xsd)

7481 The following non-normative copy of the XML schema is provided for convenience:

```

7482 (1) <?xml version="1.0" encoding="UTF-8"?>
7483 (2) <!--
7484 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
7485 (4)
7486 (5) Document number: DSP8032
7487 (6) Date: 2010-02-19
7488 (7) Version: 1.0.0
7489 (8) Document status: DMTF Standard
7490 (9)
7491 (10) Title: WS-Management Notification OperationsXML Schema
7492 (11)
7493 (12) Document type: Specification (W3C XML Schema)
7494 (13) Document language: E
7495 (14)
7496 (15) Abstract: XML Schema for WS-Management Notification Operations.
7497 (16)
7498 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
7499 (18)
7500 (19) Copyright (C) 2008-2009 Distributed Management Task Force, Inc. (DMTF).
7501 (20) All rights reserved. DMTF is a not-for-profit association of industry
7502 (21) members dedicated to promoting enterprise and systems management and
7503 (22) interoperability. Members and non-members may reproduce DMTF
7504 (23) specifications and documents, provided that correct attribution is
7505 (24) given. As DMTF specifications may be revised from time to time,
7506 (25) the particular version and release date should always be noted.
7507 (26) Implementation of certain elements of this standard or proposed
7508 (27) standard may be subject to third party patent rights, including
7509 (28) provisional patent rights (herein "patent rights"). DMTF makes
7510 (29) no representations to users of the standard as to the existence of
7511 (30) such rights, and is not responsible to recognize, disclose,
7512 (31) or identify any or all such third party patent right, owners or
7513 (32) claimants, nor for any incomplete or inaccurate identification or
7514 (33) disclosure of such rights, owners or claimants. DMTF shall have no
7515 (34) liability to any party, in any manner or circumstance, under any legal
7516 (35) theory whatsoever, for failure to recognize, disclose, or identify any
7517 (36) such third party patent rights, or for such party's reliance on the
7518 (37) standard or incorporation thereof in its product, protocols or testing
7519 (38) procedures. DMTF shall have no liability to any party implementing
7520 (39) such standard, whether such implementation is foreseeable or not, nor
7521 (40) to any patent owner or claimant, and shall have no liability or
7522 (41) responsibility for costs or losses incurred if a standard is withdrawn
7523 (42) or modified after publication, and shall be indemnified and held
7524 (43) harmless by any party implementing the standard from any and all claims
7525 (44) of infringement by a patent owner for such implementations. For
7526 (45) information about patents held by third-parties which have notified the
7527 (46) DMTF that, in their opinion, such patent may relate to or impact
7528 (47) implementations of DMTF standards, visit
7529 (48) http://www.dmtf.org/about/policies/disclosures.php.

```

```

7530 (49)
7531 (50) Change log:
7532 (51) 1.0.0 - 2009-11-01 - Work in Progress release
7533 (52) 1.0.0 - 2010-02-19 - DMTF Standard release
7534 (53)
7535 (54) -->
7536 (55) <xs:schema
7537 (56) targetNamespace="http://schemas.xmlsoap.org/ws/2004/08/eventing"
7538 (57) xmlns:tns="http://schemas.xmlsoap.org/ws/2004/08/eventing"
7539 (58) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
7540 (59) xmlns:xs="http://www.w3.org/2001/XMLSchema"
7541 (60) elementFormDefault="qualified"
7542 (61) blockDefault="#all">
7543 (62)
7544 (63) <xs:import
7545 (64) namespace="http://www.w3.org/XML/1998/namespace"
7546 (65) schemaLocation="http://www.w3.org/2001/xml.xsd" />
7547 (66) <xs:import
7548 (67) namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
7549 (68) schemaLocation="http://schemas.dmtf.org/wbem/wsman/1/DSP8034_1.0.xsd"
7550 />
7551 (69) <xs:import
7552 (70) namespace="http://www.w3.org/2005/08/addressing"
7553 (71) schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd" />
7554 (72)
7555 (73) <!-- Types and global elements -->
7556 (74) <xs:complexType name="DeliveryType" mixed="true">
7557 (75) <xs:sequence>
7558 (76) <xs:any namespace="##any" processContents="lax"
7559 (77) minOccurs="0" maxOccurs="unbounded" />
7560 (78) </xs:sequence>
7561 (79) <xs:attribute name="Mode" type="xs:anyURI" use="optional" />
7562 (80) <xs:anyAttribute namespace="##other" processContents="lax" />
7563 (81) </xs:complexType>
7564 (82)
7565 (83) <xs:simpleType name="NonNegativeDurationType">
7566 (84) <xs:restriction base="xs:duration">
7567 (85) <xs:minInclusive value="P0Y0M0DT0H0M0S" />
7568 (86) </xs:restriction>
7569 (87) </xs:simpleType>
7570 (88)
7571 (89) <xs:simpleType name="ExpirationType">
7572 (90) <xs:union memberTypes="xs:dateTime
7573 (91) tns:NonNegativeDurationType" />
7574 (92) </xs:simpleType>
7575 (93)
7576 (94) <xs:complexType name="FilterType" mixed="true">
7577 (95) <xs:sequence>
7578 (96) <xs:any namespace="##other" processContents="lax"
7579 (97) minOccurs="0" maxOccurs="unbounded" />
7580 (98) </xs:sequence>
7581 (99) <xs:attribute name="Dialect" type="xs:anyURI" use="optional" />
7582 (100) <xs:anyAttribute namespace="##other" processContents="lax" />
7583 (101) </xs:complexType>
7584 (102)
7585 (103) <xs:complexType name="LanguageSpecificStringType">
7586 (104) <xs:simpleContent>
7587 (105) <xs:extension base="xs:string">
7588 (106) <xs:attribute ref="xml:lang" />
7589 (107) <xs:anyAttribute namespace="##other" processContents="lax" />
7590 (108) </xs:extension>
7591 (109) </xs:simpleContent>
7592 (110) </xs:complexType>

```

```

7593 (111)
7594 (112) <!--
7595 (113) The type of the AnyEPRTType is effectively
7596 (114) the union of wsa04:EndpointReferenceType and
7597 (115) wsa10:EndpointReferenceType. Unfortunately, xs:union only
7598 (116) works for simple types. As a result, we have to define
7599 (117) the element in an unvalidated way to accommodate either
7600 (118) addressing type.
7601 (119) -->
7602 (120)
7603 (121) <xs:complexType name="AnyEPRTType">
7604 (122) <xs:sequence>
7605 (123) <xs:any minOccurs='1' maxOccurs='unbounded' processContents='skip'
7606 (124) namespace='##other' />
7607 (125) </xs:sequence>
7608 (126) </xs:complexType>
7609 (127)
7610 (128) <xs:element name="NotifyTo" type="tns:AnyEPRTType" />
7611 (129)
7612 (130) <!-- Subscribe request -->
7613 (131) <xs:element name="Subscribe">
7614 (132) <xs:complexType>
7615 (133) <xs:sequence>
7616 (134) <xs:element name="EndTo" type="tns:AnyEPRTType"
7617 (135) minOccurs="0" />
7618 (136) <xs:element name="Delivery" type="tns:DeliveryType" />
7619 (137) <xs:element name="Expires" type="tns:ExpirationType"
7620 (138) minOccurs="0" />
7621 (139) <xs:element name="Filter" type="tns:FilterType"
7622 (140) minOccurs="0" />
7623 (141) <xs:any namespace="##other" processContents="lax"
7624 (142) minOccurs="0" maxOccurs="unbounded" />
7625 (143) </xs:sequence>
7626 (144) <xs:anyAttribute namespace="##other" processContents="lax" />
7627 (145) </xs:complexType>
7628 (146) </xs:element>
7629 (147)
7630 (148) <xs:element name="Identifier" type="xs:anyURI" />
7631 (149)
7632 (150) <!-- Subscribe response -->
7633 (151) <xs:element name="SubscribeResponse">
7634 (152) <xs:complexType>
7635 (153) <xs:sequence>
7636 (154) <xs:element name="SubscriptionManager"
7637 (155) type="tns:AnyEPRTType" />
7638 (156) <xs:element name="Expires" type="tns:ExpirationType" />
7639 (157) <xs:any namespace="##other" processContents="lax"
7640 (158) minOccurs="0" maxOccurs="unbounded" />
7641 (159) </xs:sequence>
7642 (160) <xs:anyAttribute namespace="##other" processContents="lax" />
7643 (161) </xs:complexType>
7644 (162) </xs:element>
7645 (163)
7646 (164) <!-- Used in a fault if there's an unsupported dialect -->
7647 (165) <xs:element name="SupportedDialect" type="xs:anyURI" />
7648 (166)
7649 (167) <!-- Used in a fault if there's an unsupported delivery mode -->
7650 (168) <xs:element name="SupportedDeliveryMode" type="xs:anyURI" />
7651 (169)
7652 (170) <!-- Renew request -->
7653 (171) <xs:element name="Renew">
7654 (172) <xs:complexType>
7655 (173) <xs:sequence>

```

```

7656 (174) <xs:element name="Expires" type="tns:ExpirationType"
7657 (175) minOccurs="0" />
7658 (176) <xs:any namespace="##other" processContents="lax"
7659 (177) minOccurs="0" maxOccurs="unbounded" />
7660 (178) </xs:sequence>
7661 (179) <xs:anyAttribute namespace="##other" processContents="lax" />
7662 (180) </xs:complexType>
7663 (181) </xs:element>
7664 (182)
7665 (183) <!-- Renew response -->
7666 (184) <xs:element name="RenewResponse">
7667 (185) <xs:complexType>
7668 (186) <xs:sequence>
7669 (187) <xs:element name="Expires" type="tns:ExpirationType"
7670 (188) minOccurs="0" />
7671 (189) <xs:any namespace="##other" processContents="lax"
7672 (190) minOccurs="0" maxOccurs="unbounded" />
7673 (191) </xs:sequence>
7674 (192) <xs:anyAttribute namespace="##other" processContents="lax" />
7675 (193) </xs:complexType>
7676 (194) </xs:element>
7677 (195)
7678 (196) <!-- GetStatus request -->
7679 (197) <xs:element name="GetStatus">
7680 (198) <xs:complexType>
7681 (199) <xs:sequence>
7682 (200) <xs:any namespace="##other" processContents="lax"
7683 (201) minOccurs="0" maxOccurs="unbounded" />
7684 (202) </xs:sequence>
7685 (203) <xs:anyAttribute namespace="##other" processContents="lax" />
7686 (204) </xs:complexType>
7687 (205) </xs:element>
7688 (206)
7689 (207) <!-- GetStatus response -->
7690 (208) <xs:element name="GetStatusResponse">
7691 (209) <xs:complexType>
7692 (210) <xs:sequence>
7693 (211) <xs:element name="Expires" type="tns:ExpirationType"
7694 (212) minOccurs="0" />
7695 (213) <xs:any namespace="##other" processContents="lax"
7696 (214) minOccurs="0" maxOccurs="unbounded" />
7697 (215) </xs:sequence>
7698 (216) <xs:anyAttribute namespace="##other" processContents="lax" />
7699 (217) </xs:complexType>
7700 (218) </xs:element>
7701 (219)
7702 (220) <!-- Unsubscribe request -->
7703 (221) <xs:element name="Unsubscribe">
7704 (222) <xs:complexType>
7705 (223) <xs:sequence>
7706 (224) <xs:any namespace="##other" processContents="lax"
7707 (225) minOccurs="0" maxOccurs="unbounded" />
7708 (226) </xs:sequence>
7709 (227) <xs:anyAttribute namespace="##other" processContents="lax" />
7710 (228) </xs:complexType>
7711 (229) </xs:element>
7712 (230)
7713 (231) <!-- SubscriptionEnd message -->
7714 (232) <xs:element name="SubscriptionEnd">
7715 (233) <xs:complexType>
7716 (234) <xs:sequence>
7717 (235) <xs:element name="SubscriptionManager"
7718 (236) type="tns:AnyEPRType" />

```

```

7719 (237) <xs:element name="Status"
7720 type="tns:OpenSubscriptionEndCodeType" />
7721 (239) <xs:element name="Reason"
7722 type="tns:LanguageSpecificStringType"
7723 minOccurs="0" maxOccurs="unbounded" />
7724 (242) <xs:any namespace="##other" processContents="lax"
7725 minOccurs="0" maxOccurs="unbounded" />
7726 (244) </xs:sequence>
7727 (245) <xs:anyAttribute namespace="##other" processContents="lax" />
7728 (246) </xs:complexType>
7729 (247) </xs:element>
7730 (248)
7731 (249) <xs:simpleType name="SubscriptionEndCodeType">
7732 (250) <xs:restriction base="xs:anyURI">
7733 (251) <xs:enumeration
7734 value="http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryFailure" />
7735 (252) <xs:enumeration
7736 value="http://schemas.xmlsoap.org/ws/2004/08/eventing/SourceShuttingDown" />
7737 (253) <xs:enumeration
7738 value="http://schemas.xmlsoap.org/ws/2004/08/eventing/SourceCancelling" />
7739 (254) </xs:restriction>
7740 (255) </xs:simpleType>
7741 (256)
7742 (257) <xs:simpleType name="OpenSubscriptionEndCodeType">
7743 (258) <xs:union memberTypes="tns:SubscriptionEndCodeType xs:anyURI" />
7744 (259) </xs:simpleType>
7745 (260)
7746 (261) <xs:attribute name="EventSource" type="xs:boolean" />
7747 (262) </xs:schema>

```

7748 A normative copy of the WSDL description can be retrieved from the following address:

7749 [http://schemas.dmtf.org/wbem/wsman/1/DSP8036\\_1.0.wsdl](http://schemas.dmtf.org/wbem/wsman/1/DSP8036_1.0.wsdl)

7750 The following non-normative copy of the WSDL description is provided for convenience:

```

7751 (1) <?xml version="1.0" encoding="UTF-8"?>
7752 (2) <!--
7753 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
7754 (4)
7755 (5) Document number: DSP8036
7756 (6) Date: 2010-02-19
7757 (7) Version: 1.0.0
7758 (8) Document status: DMTF Standard
7759 (9)
7760 (10) Title: WS-Management Notification Operations WSDL
7761 (11)
7762 (12) Document type: Specification (W3C WSDL Document)
7763 (13) Document language: E
7764 (14)
7765 (15) Abstract: WSDL for WS-Management Notification Operations.
7766 (16)
7767 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
7768 (18)
7769 (19) Copyright (C) 2008-2010 Distributed Management Task Force, Inc. (DMTF).
7770 (20) All rights reserved. DMTF is a not-for-profit association of industry
7771 (21) members dedicated to promoting enterprise and systems management and
7772 (22) interoperability. Members and non-members may reproduce DMTF

```

7773 (23) specifications and documents, provided that correct attribution is  
7774 (24) given. As DMTF specifications may be revised from time to time,  
7775 (25) the particular version and release date should always be noted.  
7776 (26) Implementation of certain elements of this standard or proposed  
7777 (27) standard may be subject to third party patent rights, including  
7778 (28) provisional patent rights (herein "patent rights"). DMTF makes  
7779 (29) no representations to users of the standard as to the existence of  
7780 (30) such rights, and is not responsible to recognize, disclose,  
7781 (31) or identify any or all such third party patent right, owners or  
7782 (32) claimants, nor for any incomplete or inaccurate identification or  
7783 (33) disclosure of such rights, owners or claimants. DMTF shall have no  
7784 (34) liability to any party, in any manner or circumstance, under any legal  
7785 (35) theory whatsoever, for failure to recognize, disclose, or identify any  
7786 (36) such third party patent rights, or for such party's reliance on the  
7787 (37) standard or incorporation thereof in its product, protocols or testing  
7788 (38) procedures. DMTF shall have no liability to any party implementing  
7789 (39) such standard, whether such implementation is foreseeable or not, nor  
7790 (40) to any patent owner or claimant, and shall have no liability or  
7791 (41) responsibility for costs or losses incurred if a standard is withdrawn  
7792 (42) or modified after publication, and shall be indemnified and held  
7793 (43) harmless by any party implementing the standard from any and all claims  
7794 (44) of infringement by a patent owner for such implementations. For  
7795 (45) information about patents held by third-parties which have notified the  
7796 (46) DMTF that, in their opinion, such patent may relate to or impact  
7797 (47) implementations of DMTF standards, visit  
7798 (48) <http://www.dmtf.org/about/policies/disclosures.php>.  
7799 (49)  
7800 (50) Change log:  
7801 (51) 1.0.0 - 2009-11-01 - Work in Progress release  
7802 (52) 1.0.0.- 2010-02-19 - DMTF Standard release  
7803 (53)  
7804 (54) -->  
7805 (55) <wsdl:definitions  
7806 (56) targetNamespace="http://schemas.xmlsoap.org/ws/2004/08/eventing"  
7807 (57) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"  
7808 (58) xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"  
7809 (59) xmlns:wsme="http://schemas.xmlsoap.org/ws/2004/08/eventing"  
7810 (60) xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"  
7811 (61) xmlns:xs="http://www.w3.org/2001/XMLSchema" >  
7812 (62)  
7813 (63) <wsdl:types>  
7814 (64) <xs:schema  
7815 (65) <xs:import  
7816 (66) namespace="http://schemas.xmlsoap.org/ws/2004/08/eventing"  
7817 (67)  
7818 schemaLocation="http://schemas.dmtf.org/wbem/wsman/1/DSP8032\_1.0.xsd" />  
7819 (68) </xs:schema>  
7820 (69) </wsdl:types>  
7821 (70)  
7822 (71) <wsdl:message name="SubscribeMsg" >  
7823 (72) <wsdl:part name="body" element="wsme:Subscribe" />  
7824 (73) </wsdl:message>  
7825 (74) <wsdl:message name="SubscribeResponseMsg" >  
7826 (75) <wsdl:part name="body" element="wsme:SubscribeResponse" />  
7827 (76) </wsdl:message>  
7828 (77)  
7829 (78) <wsdl:message name="RenewMsg" >  
7830 (79) <wsdl:part name="body" element="wsme:Renew" />  
7831 (80) </wsdl:message>  
7832 (81) <wsdl:message name="RenewResponseMsg" >  
7833 (82) <wsdl:part name="body" element="wsme:RenewResponse" />  
7834 (83) </wsdl:message>  
7835 (84)

```

7836 (85) <wsdl:message name="GetStatusMsg" >
7837 (86) <wsdl:part name="body" element="wsme:GetStatus" />
7838 (87) </wsdl:message>
7839 (88) <wsdl:message name="GetStatusResponseMsg" >
7840 (89) <wsdl:part name="body" element="wsme:GetStatusResponse" />
7841 (90) </wsdl:message>
7842 (91)
7843 (92) <wsdl:message name="UnsubscribeMsg" >
7844 (93) <wsdl:part name="body" element="wsme:Unsubscribe" />
7845 (94) </wsdl:message>
7846 (95) <wsdl:message name="UnsubscribeResponseMsg" />
7847 (96)
7848 (97) <wsdl:message name="SubscriptionEnd" >
7849 (98) <wsdl:part name="body" element="wsme:SubscriptionEnd" />
7850 (99) </wsdl:message>
7851 (100)
7852 (101) <wsdl:portType name="EventSource" >
7853 (102) <wsdl:operation name="SubscribeOp" >
7854 (103) <wsdl:input
7855 (104) message="wsme:SubscribeMsg"
7856 (105)
7857 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe"
7858 (106)
7859 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe"/>
7860 (107) <wsdl:output
7861 (108) message="wsme:SubscribeResponseMsg"
7862 (109)
7863 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscribeResponse"
7864 (110)
7865 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscribeResponse"/>
7866 >
7867 (111) </wsdl:operation>
7868 (112) </wsdl:portType>
7869 (113)
7870 (114) <!-- The following portType shall be supported by the endpoint to which
7871 (115) the SubscriptionEnd message is sent. -->
7872 (116) <wsdl:portType name="EndToEndpoint">
7873 (117) <wsdl:operation name="SubscriptionEnd" >
7874 (118) <wsdl:input
7875 (119) message="wsme:SubscriptionEnd"
7876 (120)
7877 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscriptionEnd"
7878 (121)
7879 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/SubscriptionEnd"/>
7880 (122) </wsdl:operation>
7881 (123) </wsdl:portType>
7882 (124)
7883 (125) <!-- The following portType shall be supported by the endpoint to which
7884 (126) Notifications are sent. This portType also serves as a
7885 (127) mechanism by which Subscribers can know the Notifications that
7886 (128) will sent by an Event Source. -->
7887 (129) <wsdl:portType name="EventSink">
7888 (130) <!-- place the Notification messages (operations) here. For example:
7889 (131) <wsdl:operation name="WeatherReport">
7890 (132) <wsdl:input message="wr:ThunderStormMessage"
7891 (133) wsa:Action="urn:weatherReport:ThunderStorm"
7892 (134) wsam:Action="urn:weatherReport:ThunderStorm" />
7893 (135) </wsdl:operation>
7894 (136) -->
7895 (137) </wsdl:portType>
7896 (138)
7897 (139) <wsdl:portType name="SubscriptionManager" >
7898 (140) <wsdl:operation name="RenewOp" >

```



```
7899 (141) <wsdl:input
7900 (142) message="wsme:RenewMsg"
7901 (143) wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/Renew"
7902 (144)
7903 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/Renew"/>
7904 (145) <wsdl:output
7905 (146) message="wsme:RenewResponseMsg"
7906 (147)
7907 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/RenewResponse"
7908 (148)
7909 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/RenewResponse"/>
7910 (149) </wsdl:operation>
7911 (150) <wsdl:operation name="GetStatusOp" >
7912 (151) <wsdl:input
7913 (152) message="wsme:GetStatusMsg"
7914 (153)
7915 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatus"
7916 (154)
7917 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatus"/>
7918 (155) <wsdl:output
7919 (156) message="wsme:GetStatusResponseMsg"
7920 (157)
7921 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatusResponse"
7922 (158)
7923 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/GetStatusResponse"/
7924 >
7925 (159) </wsdl:operation>
7926 (160) <wsdl:operation name="UnsubscribeOp" >
7927 (161) <wsdl:input
7928 (162) message="wsme:UnsubscribeMsg"
7929 (163)
7930 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/Unsubscribe"
7931 (164)
7932 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/Unsubscribe"/>
7933 (165) <wsdl:output
7934 (166) message="wsme:UnsubscribeResponseMsg"
7935 (167)
7936 wsa:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/UnsubscribeResponse"
7937 (168)
7938 wsam:Action="http://schemas.xmlsoap.org/ws/2004/08/eventing/UnsubscribeResponse"
7939 "/>
7940 (169) </wsdl:operation>
7941 (170) </wsdl:portType>
7942 (171) </wsdl:definitions>
```

7943

## ANNEX J (informative)

7944  
7945  
7946  
7947

### Addressing XML Schema

7948 A normative copy of the XML schemas for the addressing features can be retrieved at the following  
7949 address:

7950 [http://schemas.dmtf.org/wbem/wsman/1/DSP8034\\_1.0.xsd](http://schemas.dmtf.org/wbem/wsman/1/DSP8034_1.0.xsd)

7951 The following non-normative copy of the XML schema is provided for convenience:

```
7952 (1) <?xml version="1.0" encoding="UTF-8"?>
7953 (2) <!--
7954 (3) DMTF - Distributed Management Task Force, Inc. - http://www.dmtf.org
7955 (4)
7956 (5) Document number: DSP8034
7957 (6) Date: 2010-02-19
7958 (7) Version: 1.0.0
7959 (8) Document status: DMTF Standard
7960 (9)
7961 (10) Title: WS-Management Addressing XML Schema
7962 (11)
7963 (12) Document type: Specification (W3C XML Schema)
7964 (13) Document language: E
7965 (14)
7966 (15) Abstract: XML Schema for WS-Management Addressing.
7967 (16)
7968 (17) Contact group: DMTF WS-Management Work Group, wsman-chair@dmtof.org
7969 (18)
7970 (19) Copyright (C) 2008-2010 Distributed Management Task Force, Inc. (DMTF).
7971 (20) All rights reserved. DMTF is a not-for-profit association of industry
7972 (21) members dedicated to promoting enterprise and systems management and
7973 (22) interoperability. Members and non-members may reproduce DMTF
7974 (23) specifications and documents, provided that correct attribution is
7975 (24) given. As DMTF specifications may be revised from time to time,
7976 (25) the particular version and release date should always be noted.
7977 (26) Implementation of certain elements of this standard or proposed
7978 (27) standard may be subject to third party patent rights, including
7979 (28) provisional patent rights (herein "patent rights"). DMTF makes
7980 (29) no representations to users of the standard as to the existence of
7981 (30) such rights, and is not responsible to recognize, disclose,
7982 (31) or identify any or all such third party patent right, owners or
7983 (32) claimants, nor for any incomplete or inaccurate identification or
7984 (33) disclosure of such rights, owners or claimants. DMTF shall have no
7985 (34) liability to any party, in any manner or circumstance, under any legal
7986 (35) theory whatsoever, for failure to recognize, disclose, or identify any
7987 (36) such third party patent rights, or for such party's reliance on the
7988 (37) standard or incorporation thereof in its product, protocols or testing
7989 (38) procedures. DMTF shall have no liability to any party implementing
7990 (39) such standard, whether such implementation is foreseeable or not, nor
7991 (40) to any patent owner or claimant, and shall have no liability or
7992 (41) responsibility for costs or losses incurred if a standard is withdrawn
7993 (42) or modified after publication, and shall be indemnified and held
7994 (43) harmless by any party implementing the standard from any and all claims
7995 (44) of infringement by a patent owner for such implementations. For
7996 (45) information about patents held by third-parties which have notified the
7997 (46) DMTF that, in their opinion, such patent may relate to or impact
7998 (47) implementations of DMTF standards, visit
7999 (48) http://www.dmtf.org/about/policies/disclosures.php.
```

```

8000 (49)
8001 (50) Change log:
8002 (51) 1.0.0 - 2009-11-01 - Work in Progress release
8003 (52) 1.0.0 - 2010-02-19 - DMTF Standard release
8004 (53) -->
8005 (54) <xs:schema
8006 (55) targetNamespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
8007 (56) xmlns:xs="http://www.w3.org/2001/XMLSchema"
8008 (57) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
8009 (58) elementFormDefault="qualified" blockDefault="#all">
8010 (59)
8011 (60) <!-- // Addressing ////////////////////////////////// -->
8012 (61) <!-- Endpoint reference -->
8013 (62) <xs:element name="EndpointReference" type="wsa:EndpointReferenceType"/>
8014 (63) <xs:complexType name="EndpointReferenceType">
8015 (64) <xs:sequence>
8016 (65) <xs:element name="Address" type="wsa:AttributedURI"/>
8017 (66) <xs:element name="ReferenceProperties"
8018 (67) type="wsa:ReferencePropertiesType" minOccurs="0"/>
8019 (68) <xs:element name="ReferenceParameters"
8020 (69) type="wsa:ReferenceParametersType" minOccurs="0"/>
8021 (70) <xs:element name="PortType" type="wsa:AttributedQName"
8022 (71) minOccurs="0"/>
8023 (71) <xs:element name="ServiceName" type="wsa:ServiceNameType"
8024 (72) minOccurs="0"/>
8025 (72) <xs:any namespace="##other" processContents="lax" minOccurs="0"
8026 (73) maxOccurs="unbounded">
8027 (74) <xs:annotation>
8028 (75) <xs:documentation>
8029 (76) If "Policy" elements from namespace
8030 (77) "http://schemas.xmlsoap.org/ws/2002/12/policy#policy" are used,
8031 (78) they must appear first (before any extensibility elements).
8032 (79) </xs:documentation>
8033 (80) </xs:annotation>
8034 (81) </xs:any>
8035 (82) </xs:sequence>
8036 (83) <xs:anyAttribute namespace="##other" processContents="lax"/>
8037 (84) </xs:complexType>
8038 (85) <xs:complexType name="ReferencePropertiesType">
8039 (86) <xs:sequence>
8040 (87) <xs:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
8041 (88) </xs:sequence>
8042 (89) </xs:complexType>
8043 (90) <xs:complexType name="ReferenceParametersType">
8044 (91) <xs:sequence>
8045 (92) <xs:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
8046 (93) </xs:sequence>
8047 (94) </xs:complexType>
8048 (95) <xs:complexType name="ServiceNameType">
8049 (96) <xs:simpleContent>
8050 (97) <xs:extension base="xs:QName">
8051 (98) <xs:attribute name="PortName" type="xs:NCName"/>
8052 (99) <xs:anyAttribute namespace="##other" processContents="lax"/>
8053 (100) </xs:extension>
8054 (101) </xs:simpleContent>
8055 (102) </xs:complexType>
8056 (103) <!-- Message information header blocks -->
8057 (104) <xs:element name="MessageID" type="wsa:AttributedURI"/>
8058 (105) <xs:element name="RelatesTo" type="wsa:Relationship"/>
8059 (106) <xs:element name="To" type="wsa:AttributedURI"/>
8060 (107) <xs:element name="Action" type="wsa:AttributedURI"/>
8061 (108) <xs:element name="From" type="wsa:EndpointReferenceType"/>
8062 (109) <xs:element name="ReplyTo" type="wsa:EndpointReferenceType"/>

```

```

8063 (110) <xs:element name="FaultTo" type="wsa:EndpointReferenceType"/>
8064 (111) <xs:complexType name="Relationship">
8065 (112) <xs:simpleContent>
8066 (113) <xs:extension base="xs:anyURI">
8067 (114) <xs:attribute name="RelationshipType" type="xs:QName"
8068 use="optional"/>
8069 (115) <xs:anyAttribute namespace="##other" processContents="lax"/>
8070 (116) </xs:extension>
8071 (117) </xs:simpleContent>
8072 (118) </xs:complexType>
8073 (119) <xs:simpleType name="RelationshipTypeValues">
8074 (120) <xs:restriction base="xs:QName">
8075 (121) <xs:enumeration value="wsa:Reply"/>
8076 (122) </xs:restriction>
8077 (123) </xs:simpleType>
8078 (124) <xs:element name="ReplyAfter" type="wsa:ReplyAfterType"/>
8079 (125) <xs:complexType name="ReplyAfterType">
8080 (126) <xs:simpleContent>
8081 (127) <xs:extension base="xs:nonNegativeInteger">
8082 (128) <xs:anyAttribute namespace="##other"/>
8083 (129) </xs:extension>
8084 (130) </xs:simpleContent>
8085 (131) </xs:complexType>
8086 (132) <xs:element name="RetryAfter" type="wsa:RetryAfterType"/>
8087 (133) <xs:complexType name="RetryAfterType">
8088 (134) <xs:simpleContent>
8089 (135) <xs:extension base="xs:nonNegativeInteger">
8090 (136) <xs:anyAttribute namespace="##other"/>
8091 (137) </xs:extension>
8092 (138) </xs:simpleContent>
8093 (139) </xs:complexType>
8094 (140) <xs:simpleType name="FaultSubcodeValues">
8095 (141) <xs:restriction base="xs:QName">
8096 (142) <xs:enumeration value="wsa:InvalidMessageInformationHeader"/>
8097 (143) <xs:enumeration value="wsa:MessageInformationHeaderRequired"/>
8098 (144) <xs:enumeration value="wsa:DestinationUnreachable"/>
8099 (145) <xs:enumeration value="wsa:ActionNotSupported"/>
8100 (146) <xs:enumeration value="wsa:EndpointUnavailable"/>
8101 (147) </xs:restriction>
8102 (148) </xs:simpleType>
8103 (149) <xs:attribute name="Action" type="xs:anyURI"/>
8104 (150) <!-- Common declarations and definitions -->
8105 (151) <xs:complexType name="AttributedQName">
8106 (152) <xs:simpleContent>
8107 (153) <xs:extension base="xs:QName">
8108 (154) <xs:anyAttribute namespace="##other" processContents="lax"/>
8109 (155) </xs:extension>
8110 (156) </xs:simpleContent>
8111 (157) </xs:complexType>
8112 (158) <xs:complexType name="AttributedURI">
8113 (159) <xs:simpleContent>
8114 (160) <xs:extension base="xs:anyURI">
8115 (161) <xs:anyAttribute namespace="##other" processContents="lax"/>
8116 (162) </xs:extension>
8117 (163) </xs:simpleContent>
8118 (164) </xs:complexType>
8119 (165) </xs:schema>

```

8120  
8121  
8122  
8123

## ANNEX K (informative)

### WS-Management XML Schema

8124 A normative copy of the XML schemas for WS-Management can be retrieved at the following  
8125 address:

8126 <http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd>

8127 The following non-normative copy of the XML schema is provided for convenience:

```
8128 (1) <?xml version="1.0" encoding="UTF-8"?>
8129 (2) <!--
8130 (3) Notice
8131 (4) DSP8015
8132 (5) Document: WS-Management protocol XML Schema
8133 (6) Version: 1.0.0
8134 (7) Status: Final
8135 (8) Date: 01/20/2008
8136 (9) Author: Bryan Murray, et al.
8137 (10) Description: XML Schema for WS-Management protocol
8138 (11)
8139 (12) Copyright © 2008 Distributed Management Task Force, Inc. (DMTF). All rights
8140 reserved. DMTF is a not-for-profit association of industry members dedicated to
8141 promoting enterprise and systems management and interoperability. Members and
8142 non-members may reproduce DMTF specifications and documents, provided that
8143 correct attribution is given. As DMTF specifications may be revised from time
8144 to time, the particular version and release date should always be noted.
8145 Implementation of certain elements of this standard or proposed standard may be
8146 subject to third party patent rights, including provisional patent rights
8147 (herein "patent rights"). DMTF makes no representations to users of the
8148 standard as to the existence of such rights, and is not responsible to
8149 recognize, disclose, or identify any or all such third party patent right,
8150 owners or claimants, nor for any incomplete or inaccurate identification or
8151 disclosure of such rights, owners or claimants. DMTF shall have no liability to
8152 any party, in any manner or circumstance, under any legal theory whatsoever,
8153 for failure to recognize, disclose, or identify any such third party patent
8154 rights, or for such party's reliance on the standard or incorporation thereof
8155 in its product, protocols or testing procedures. DMTF shall have no liability
8156 to any party implementing such standard, whether such implementation is
8157 foreseeable or not, nor to any patent owner or claimant, and shall have no
8158 liability or responsibility for costs or losses incurred if a standard is
8159 withdrawn or modified after publication, and shall be indemnified and held
8160 harmless by any party implementing the standard from any and all claims of
8161 infringement by a patent owner for such implementations. For information about
8162 patents held by third-parties which have notified the DMTF that, in their
8163 opinion, such patent may relate to or impact implementations of DMTF standards,
8164 visit http://www.dmtf.org/about/policies/disclosures.php.
8165 (13)
8166 (14) Change Requests:
8167 (15) None
8168 (16) -->
8169 (17) <xs:schema targetNamespace="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
8170 (18) xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
8171 (19) xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
8172 (20) xmlns:xs="http://www.w3.org/2001/XMLSchema"
8173 (21) elementFormDefault="qualified" version="1.0.0e">
8174 (22)
```

```

8175 (23) <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
8176 (24)
8177 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
8178 (25) <xs:import namespace="http://www.w3.org/XML/1998/namespace"
8179 (26) schemaLocation="http://www.w3.org/2001/xml.xsd"/>
8180 (27)
8181 (28) <xs:complexType name="attributableURI">
8182 (29) <xs:simpleContent>
8183 (30) <xs:extension base="xs:anyURI">
8184 (31) <xs:anyAttribute namespace="##other" processContents="lax"/>
8185 (32) </xs:extension>
8186 (33) </xs:simpleContent>
8187 (34) </xs:complexType>
8188 (35)
8189 (36) <xs:element name="ResourceURI" type="wsman:attributableURI"/>
8190 (37)
8191 (38) <xs:complexType name="SelectorType">
8192 (39) <xs:annotation>
8193 (40) <xs:documentation>
8194 (41) Instances of this type can be only simple types or EPRs, not
8195 (42) arbitrary mixed data.
8196 (42) </xs:documentation>
8197 (43) </xs:annotation>
8198 (44) <xs:complexContent mixed="true">
8199 (45) <xs:restriction base="xs:anyType">
8200 (46) <xs:sequence>
8201 (47) <xs:element ref="wsa:EndpointReference" minOccurs="0"/>
8202 (48) </xs:sequence>
8203 (49) <xs:attribute name="Name" type="xs:NCName" use="required"/>
8204 (50) <xs:anyAttribute namespace="##other" processContents="lax"/>
8205 (51) </xs:restriction>
8206 (52) </xs:complexContent>
8207 (53) </xs:complexType>
8208 (54) <xs:element name="Selector" type="wsman:SelectorType"/>
8209 (55)
8210 (56) <xs:complexType name="SelectorSetType">
8211 (57) <xs:sequence>
8212 (58) <xs:element ref="wsman:Selector" minOccurs="1" maxOccurs="unbounded"/>
8213 (59) </xs:sequence>
8214 (60) <xs:anyAttribute namespace="##other" processContents="lax"/>
8215 (61) </xs:complexType>
8216 (62)
8217 (63) <xs:element name="SelectorSet" type="wsman:SelectorSetType">
8218 (64) <xs:unique name="oneSelectorPerName">
8219 (65) <xs:selector xpath="./Selector"/>
8220 (66) <xs:field xpath="@Name"/>
8221 (67) </xs:unique>
8222 (68) </xs:element>
8223 (69)
8224 (70) <xs:complexType name="attributableDuration">
8225 (71) <xs:simpleContent>
8226 (72) <xs:extension base="xs:duration">
8227 (73) <xs:anyAttribute namespace="##other" processContents="lax"/>
8228 (74) </xs:extension>
8229 (75) </xs:simpleContent>
8230 (76) </xs:complexType>
8231 (77)
8232 (78) <xs:element name="OperationTimeout" type="wsman:attributableDuration"/>
8233 (79)
8234 (80) <xs:complexType name="attributablePositiveInteger">

```

```

8235 (81) <xs:simpleContent>
8236 (82) <xs:extension base="xs:positiveInteger">
8237 (83) <xs:anyAttribute namespace="##other" processContents="lax"/>
8238 (84) </xs:extension>
8239 (85) </xs:simpleContent>
8240 (86) </xs:complexType>
8241 (87)
8242 (88) <xs:simpleType name="PolicyType">
8243 (89) <xs:restriction base="xs:token">
8244 (90) <xs:enumeration value="CancelSubscription"/>
8245 (91) <xs:enumeration value="Skip"/>
8246 (92) <xs:enumeration value="Notify"/>
8247 (93) </xs:restriction>
8248 (94) </xs:simpleType>
8249 (95)
8250 (96) <xs:complexType name="MaxEnvelopeSizeType">
8251 (97) <xs:simpleContent>
8252 (98) <xs:extension base="wsman:attributablePositiveInteger">
8253 (99) <xs:attribute name="Policy" type="wsman:PolicyType"
8254 default="Notify"/>
8255 (100) </xs:extension>
8256 (101) </xs:simpleContent>
8257 (102) </xs:complexType>
8258 (103) <xs:element name="MaxEnvelopeSize" type="wsman:MaxEnvelopeSizeType"/>
8259 (104)
8260 (105) <xs:element name="Locale">
8261 (106) <xs:complexType>
8262 (107) <xs:attribute ref="xml:lang" use="required"/>
8263 (108) <xs:anyAttribute namespace="##other" processContents="lax"/>
8264 (109) </xs:complexType>
8265 (110) </xs:element>
8266 (111)
8267 (112) <xs:complexType name="OptionType">
8268 (113) <xs:simpleContent>
8269 (114) <xs:extension base="xs:string">
8270 (115) <xs:attribute name="Name" type="xs:NCName" use="required"/>
8271 (116) <xs:attribute name="MustComply" type="xs:boolean" default="false"/>
8272 (117) <xs:attribute name="Type" type="xs:QName"/>
8273 (118) <xs:anyAttribute namespace="##other" processContents="lax"/>
8274 (119) </xs:extension>
8275 (120) </xs:simpleContent>
8276 (121) </xs:complexType>
8277 (122) <xs:element name="Option" type="wsman:OptionType"/>
8278 (123)
8279 (124) <xs:element name="OptionSet">
8280 (125) <xs:complexType>
8281 (126) <xs:sequence>
8282 (127) <xs:element ref="wsman:Option" minOccurs="0" maxOccurs="unbounded"/>
8283 (128) </xs:sequence>
8284 (129) <xs:anyAttribute namespace="##other" processContents="lax"/>
8285 (130) </xs:complexType>
8286 (131) </xs:element>
8287 (132)
8288 (133) <xs:complexType name="attributableEmpty">
8289 (134) <xs:anyAttribute namespace="##other" processContents="lax"/>
8290 (135) </xs:complexType>
8291 (136)
8292 (137) <xs:element name="RequestEPR" type="wsman:attributableEmpty"/>
8293 (138) <xs:element name="EPRInvalid" type="wsman:attributableEmpty"/>
8294 (139) <xs:element name="EPRUnknown" type="wsman:attributableEmpty"/>

```

```

8295 (140)
8296 (141) <xs:complexType name="RequestedEPRTYPE">
8297 (142) <xs:choice>
8298 (143) <xs:element ref="wsa:EndpointReference"/>
8299 (144) <xs:element ref="wsman:EPRInvalid"/>
8300 (145) <xs:element ref="wsman:EPRUnknown"/>
8301 (146) </xs:choice>
8302 (147) <xs:anyAttribute namespace="##other" processContents="lax"/>
8303 (148) </xs:complexType>
8304 (149) <xs:element name="RequestedEPR" type="wsman:RequestedEPRTYPE"/>
8305 (150)
8306 (151) <xs:complexType name="mixedDataType">
8307 (152) <xs:complexContent mixed="true">
8308 (153) <xs:restriction base="xs:anyType">
8309 (154) <xs:sequence>
8310 (155) <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
8311 processContents="skip"/>
8312 (156) </xs:sequence>
8313 (157) <xs:anyAttribute namespace="##other" processContents="lax"/>
8314 (158) </xs:restriction>
8315 (159) </xs:complexContent>
8316 (160) </xs:complexType>
8317 (161)
8318 (162) <xs:complexType name="fragmentMixedDataType">
8319 (163) <xs:complexContent mixed="true">
8320 (164) <xs:extension base="wsman:mixedDataType">
8321 (165) <xs:attribute name="Dialect" type="xs:anyURI"
8322 default="http://www.w3.org/TR/1999/REC-xpath-19991116"/>
8323 (166) <xs:anyAttribute namespace="##other" processContents="lax"/>
8324 (167) </xs:extension>
8325 (168) </xs:complexContent>
8326 (169) </xs:complexType>
8327 (170)
8328 (171) <xs:element name="FragmentTransfer" type="wsman:fragmentMixedDataType"/>
8329 (172) <xs:element name="XmlFragment" type="wsman:mixedDataType"/>
8330 (173)
8331 (174) <xs:complexType name="attributableNonNegativeInteger">
8332 (175) <xs:simpleContent>
8333 (176) <xs:extension base="xs:nonNegativeInteger">
8334 (177) <xs:anyAttribute namespace="##other" processContents="lax"/>
8335 (178) </xs:extension>
8336 (179) </xs:simpleContent>
8337 (180) </xs:complexType>
8338 (181)
8339 (182) <xs:element name="TotalItemsCountEstimate"
8340 type="wsman:attributableNonNegativeInteger" nillable="true"/>
8341 (183) <xs:element name="RequestTotalItemsCountEstimate"
8342 type="wsman:attributableEmpty"/>
8343 (184)
8344 (185) <xs:element name="OptimizeEnumeration" type="wsman:attributableEmpty"/>
8345 (186) <xs:element name="MaxElements" type="wsman:attributablePositiveInteger"/>
8346 (187)
8347 (188) <xs:simpleType name="EnumerationModeType">
8348 (189) <xs:restriction base="xs:token">
8349 (190) <xs:enumeration value="EnumerateEPR"/>
8350 (191) <xs:enumeration value="EnumerateObjectAndEPR"/>
8351 (192) </xs:restriction>
8352 (193) </xs:simpleType>
8353 (194) <xs:element name="EnumerationMode" type="wsman:EnumerationModeType"/>
8354 (195)

```



```

8355 (196) <xs:complexType name="mixedDataFilterType" mixed="true">
8356 (197) <xs:complexContent mixed="true">
8357 (198) <xs:restriction base="xs:anyType">
8358 (199) <xs:sequence>
8359 (200) <xs:any namespace="##any" processContents="skip" minOccurs="0"
8360 maxOccurs="unbounded"/>
8361 (201) </xs:sequence>
8362 (202) <xs:anyAttribute namespace="##any" processContents="lax"/>
8363 (203) </xs:restriction>
8364 (204) </xs:complexContent>
8365 (205) </xs:complexType>
8366 (206)
8367 (207) <xs:complexType name="filterMixedDataType" mixed="true">
8368 (208) <xs:complexContent mixed="true">
8369 (209) <xs:extension base="wsman:mixedDataFilterType">
8370 (210) <xs:attribute name="Dialect" type="xs:anyURI"
8371 default="http://www.w3.org/TR/1999/REC-xpath-19991116"/>
8372 (211) <xs:anyAttribute namespace="##any" processContents="lax"/>
8373 (212) </xs:extension>
8374 (213) </xs:complexContent>
8375 (214) </xs:complexType>
8376 (215)
8377 (216) <xs:element name="Filter" type="wsman:filterMixedDataType"/>
8378 (217)
8379 (218) <xs:complexType name="ObjectAndEPRTType">
8380 (219) <xs:sequence>
8381 (220) <xs:any namespace="##any" processContents="lax"/>
8382 (221) <xs:element ref="wsa:EndpointReference"/>
8383 (222) </xs:sequence>
8384 (223) </xs:complexType>
8385 (224) <xs:element name="Item" type="wsman:ObjectAndEPRTType"/>
8386 (225)
8387 (226) <xs:complexType name="anyListType">
8388 (227) <xs:sequence>
8389 (228) <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
8390 processContents="lax"/>
8391 (229) </xs:sequence>
8392 (230) <xs:anyAttribute namespace="##other" processContents="lax"/>
8393 (231) </xs:complexType>
8394 (232)
8395 (233) <xs:element name="Items" type="wsman:anyListType"/>
8396 (234) <xs:element name="EndOfSequence" type="wsman:attributableEmpty"/>
8397 (235)
8398 (236) <xs:complexType name="attributableLanguage">
8399 (237) <xs:simpleContent>
8400 (238) <xs:extension base="xs:language">
8401 (239) <xs:anyAttribute namespace="##other" processContents="lax"/>
8402 (240) </xs:extension>
8403 (241) </xs:simpleContent>
8404 (242) </xs:complexType>
8405 (243)
8406 (244) <xs:element name="ContentEncoding" type="wsman:attributableLanguage"/>
8407 (245)
8408 (246) <xs:complexType name="ConnectionRetryType">
8409 (247) <xs:simpleContent>
8410 (248) <xs:extension base="wsman:attributableDuration">
8411 (249) <xs:attribute name="Total" type="xs:unsignedLong"/>
8412 (250) </xs:extension>
8413 (251) </xs:simpleContent>
8414 (252) </xs:complexType>

```

```

8415 (253) <xs:element name="ConnectionRetry" type="wsman:ConnectionRetryType"/>
8416 (254)
8417 (255) <xs:element name="Heartbeats" type="wsman:attributableDuration"/>
8418 (256) <xs:element name="SendBookmarks" type="wsman:attributableEmpty"/>
8419 (257)
8420 (258) <xs:complexType name="attributableAny">
8421 (259) <xs:sequence>
8422 (260) <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
8423 processContents="lax"/>
8424 (261) </xs:sequence>
8425 (262) <xs:anyAttribute namespace="##other" processContents="lax"/>
8426 (263) </xs:complexType>
8427 (264)
8428 (265) <xs:element name="Bookmark" type="wsman:mixedDataType"/>
8429 (266) <xs:element name="MaxTime" type="wsman:attributableDuration"/>
8430 (267)
8431 (268) <xs:complexType name="EventType">
8432 (269) <xs:complexContent>
8433 (270) <xs:extension base="wsman:attributableAny">
8434 (271) <xs:attribute name="Action" type="xs:anyURI" use="required"/>
8435 (272) </xs:extension>
8436 (273) </xs:complexContent>
8437 (274) </xs:complexType>
8438 (275) <xs:element name="Event" type="wsman:EventType"/>
8439 (276)
8440 (277) <xs:complexType name="EventsType">
8441 (278) <xs:sequence>
8442 (279) <xs:element ref="wsman:Event" minOccurs="1" maxOccurs="unbounded"/>
8443 (280) </xs:sequence>
8444 (281) <xs:anyAttribute namespace="##other" processContents="lax"/>
8445 (282) </xs:complexType>
8446 (283) <xs:element name="Events" type="wsman:EventsType"/>
8447 (284)
8448 (285) <xs:element name="AckRequested" type="wsman:attributableEmpty"/>
8449 (286)
8450 (287) <xs:complexType name="attributableInt">
8451 (288) <xs:simpleContent>
8452 (289) <xs:extension base="xs:int">
8453 (290) <xs:anyAttribute namespace="##other" processContents="lax"/>
8454 (291) </xs:extension>
8455 (292) </xs:simpleContent>
8456 (293) </xs:complexType>
8457 (294)
8458 (295) <xs:complexType name="DroppedEventsType">
8459 (296) <xs:simpleContent>
8460 (297) <xs:extension base="wsman:attributableInt">
8461 (298) <xs:attribute name="Action" type="xs:anyURI" use="required"/>
8462 (299) </xs:extension>
8463 (300) </xs:simpleContent>
8464 (301) </xs:complexType>
8465 (302) <xs:element name="DroppedEvents" type="wsman:DroppedEventsType"/>
8466 (303)
8467 (304) <xs:simpleType name="restrictedProfileType">
8468 (305) <xs:restriction base="xs:anyURI">
8469 (306) <xs:enumeration
8470 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/basic"/>
8471 (307) <xs:enumeration
8472 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/digest"/>
8473 (308) <xs:enumeration
8474 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/basic"/>

```

```

8475 (309) <xs:enumeration
8476 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/digest"/>
8477 (310) <xs:enumeration
8478 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual"/>
8479 (311) <xs:enumeration
8480 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/basic
8481 "/>
8482 (312) <xs:enumeration
8483 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/diges
8484 t"/>
8485 (313) <xs:enumeration
8486 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/spnego-
8487 kerberos"/>
8488 (314) <xs:enumeration
8489 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/https/mutual/spneg
8490 o-kerberos"/>
8491 (315) <xs:enumeration
8492 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/secprofile/http/spnego-
8493 kerberos"/>
8494 (316) </xs:restriction>
8495 (317) </xs:simpleType>
8496 (318)
8497 (319) <xs:simpleType name="ProfileType">
8498 (320) <xs:union memberTypes="wsman:restrictedProfileType xs:anyURI"/>
8499 (321) </xs:simpleType>
8500 (322)
8501 (323) <xs:complexType name="AuthType">
8502 (324) <xs:complexContent>
8503 (325) <xs:extension base="wsman:attributableEmpty">
8504 (326) <xs:attribute name="Profile" type="wsman:ProfileType"
8505 use="required"/>
8506 (327) </xs:extension>
8507 (328) </xs:complexContent>
8508 (329) </xs:complexType>
8509 (330) <xs:element name="Auth" type="wsman:AuthType"/>
8510 (331)
8511 (332) <xs:simpleType name="ThumbprintType">
8512 (333) <xs:restriction base="xs:string">
8513 (334) <xs:pattern value="[0-9a-fA-F]{40}"/>
8514 (335) </xs:restriction>
8515 (336) </xs:simpleType>
8516 (337) <xs:element name="CertificateThumbprint" type="wsman:ThumbprintType"/>
8517 (338)
8518 (339)
8519 (340) <xs:simpleType name="restrictedFaultDetailType">
8520 (341) <xs:restriction base="xs:anyURI">
8521 (342) <xs:enumeration
8522 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ActionMismatch"/>
8523 (343) <xs:enumeration
8524 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Ack"/>
8525 (344) <xs:enumeration
8526 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AddressingMode"/>
8527 (345) <xs:enumeration
8528 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/AsynchronousReque
8529 st"/>
8530 (346) <xs:enumeration
8531 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Bookmarks"/>
8532 (347) <xs:enumeration
8533 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/CharacterSet"/>
8534 (348) <xs:enumeration
8535 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DeliveryRetries"/

```

```

8536 >
8537 (349) <xs:enumeration
8538 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/DuplicateSelector
8539 s"/>
8540 (350) <xs:enumeration
8541 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EncodingType"/>
8542 (351) <xs:enumeration
8543 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/EnumerationMode"/
8544 >
8545 (352) <xs:enumeration
8546 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ExpirationTime"/>
8547 (353) <xs:enumeration
8548 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Expired"/>
8549 (354) <xs:enumeration
8550 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FilteringRequired
8551 "/>
8552 (355) <xs:enumeration
8553 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FormatMismatch"/>
8554 (356) <xs:enumeration
8555 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/FragmentLevelAcce
8556 ss"/>
8557 (357) <xs:enumeration
8558 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Heartbeats"/>
8559 (358) <xs:enumeration
8560 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsecureAddress"/
8561 >
8562 (359) <xs:enumeration
8563 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InsufficientSelec
8564 tors"/>
8565 (360) <xs:enumeration
8566 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Invalid"/>
8567 (361) <xs:enumeration
8568 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidName"/>
8569 (362) <xs:enumeration
8570 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidFragment"/
8571 >
8572 (363) <xs:enumeration
8573 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidNamespace"
8574 />
8575 (364) <xs:enumeration
8576 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidResourceUR
8577 I"/>
8578 (365) <xs:enumeration
8579 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValue"/>
8580 (366) <xs:enumeration
8581 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidValues"/>
8582 (367) <xs:enumeration
8583 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Locale"/>
8584 (368) <xs:enumeration
8585 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxElements"/>
8586 (369) <xs:enumeration
8587 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopePolicy
8588 "/>
8589 (370) <xs:enumeration
8590 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxEnvelopeSize"/
8591 >
8592 (371) <xs:enumeration
8593 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MaxTime"/>
8594 (372) <xs:enumeration
8595 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MinimumEnvelopeLi
8596 mit"/>
8597 (373) <xs:enumeration

```

```

8598 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/MissingValues"/>
8599 (374) <xs:enumeration
8600 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/NotSupported"/>
8601 (375) <xs:enumeration
8602 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OperationTimeout"
8603 />
8604 (376) <xs:enumeration
8605 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/OptionLimit"/>
8606 (377) <xs:enumeration
8607 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ResourceOffline"/
8608 >
8609 (378) <xs:enumeration
8610 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/SelectorLimit"/>
8611 (379) <xs:enumeration
8612 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/ServiceEnvelopeLi
8613 mit"/>
8614 (380) <xs:enumeration
8615 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/TypeMismatch"/>
8616 (381) <xs:enumeration
8617 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnexpectedSelecto
8618 rs"/>
8619 (382) <xs:enumeration
8620 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnreportableSucce
8621 ss"/>
8622 (383) <xs:enumeration
8623 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnsupportedCharac
8624 ter"/>
8625 (384) <xs:enumeration
8626 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnusableAddress"/
8627 >
8628 (385) <xs:enumeration
8629 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/URILimitExceeded"
8630 />
8631 (386) <xs:enumeration
8632 value="http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/Whitespace"/>
8633 (387) </xs:restriction>
8634 (388) </xs:simpleType>
8635 (389)
8636 (390) <xs:simpleType name="FaultDetailType">
8637 (391) <xs:union memberTypes="wsman:restrictedFaultDetailType xs:anyURI"/>
8638 (392) </xs:simpleType>
8639 (393)
8640 (394) <xs:element name="FaultDetail" type="wsman:FaultDetailType"/>
8641 (395) <xs:element name="FragmentDialect" type="wsman:attributableURI"/>
8642 (396) <xs:element name="SupportedSelectorName" type="xs:NCName"/>
8643 (397)
8644 (398) <!-- Master Fault Table subcode QNames -->
8645 (399) <xs:element name="AccessDenied"><xs:complexType/></xs:element>
8646 (400) <xs:element name="AlreadyExists"><xs:complexType/></xs:element>
8647 (401) <xs:element name="CannotProcessFilter"><xs:complexType/></xs:element>
8648 (402) <xs:element name="Concurrency"><xs:complexType/></xs:element>
8649 (403) <xs:element name="DeliveryRefused"><xs:complexType/></xs:element>
8650 (404) <xs:element name="EncodingLimit"><xs:complexType/></xs:element>
8651 (405) <xs:element name="EventDeliverToUnusable"><xs:complexType/></xs:element>
8652 (406) <xs:element
8653 name="FragmentDialectNotSupported"><xs:complexType/></xs:element>
8654 (407) <xs:element name="InternalServerError"><xs:complexType/></xs:element>
8655 (408) <xs:element name="InvalidBookmark"><xs:complexType/></xs:element>
8656 (409) <xs:element name="InvalidOptions"><xs:complexType/></xs:element>
8657 (410) <xs:element name="InvalidParameter"><xs:complexType/></xs:element>
8658 (411) <xs:element name="InvalidSelectors"><xs:complexType/></xs:element>

```

```
8659 (412) <xs:element name="NoAck"><xs:complexType/></xs:element>
8660 (413) <xs:element name="QuotaLimit"><xs:complexType/></xs:element>
8661 (414) <xs:element name="SchemaValidationError"><xs:complexType/></xs:element>
8662 (415) <xs:element name="TimedOut"><xs:complexType/></xs:element>
8663 (416) <xs:element name="UnsupportedFeature"><xs:complexType/></xs:element>
8664 (417)
8665 (418) </xs:schema>
```

8666

8667  
8668  
8669  
8670

## ANNEX L (informative)

### Change Log

8671

| Version | Date       | Description                                                                                                                                                                                                                  |
|---------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.0.0   | 2008-02-12 | Released as Final Standard                                                                                                                                                                                                   |
| 1.1.0   | 2010-03-03 | Released as DMTF Standard, with the following changes: <ul style="list-style-type: none"><li>• Incorporates TEEN specifications inline</li><li>• Addresses consistency issues with DSP0227 on Put and Fragment Put</li></ul> |
| 1.1.1   | 2012-07-30 | Incorporate additional clarifying text to Forward section for ISO/IEC publication as Publicly Available Specification (PAS)                                                                                                  |
| 1.1.1   | 2012-08-28 | DMTF Standard                                                                                                                                                                                                                |
| 1.2.0a  | 2014-06-16 | wgv 0.8.0 Release as DMTF Work In Progress                                                                                                                                                                                   |
| 1.2.0b  | 2014-06-16 | wgv 0.9.0 Release as DMTF Standard                                                                                                                                                                                           |

8672