



1

2

3

4

Document Number: DSP0227

Date: 2009-06-19

Version: 1.0.0

5

WS-Management CIM Binding Specification

6

Document Type: Specification

7

Document Status: DMTF Standard

8

Document Language: E

9

10 Copyright Notice

11 Copyright © 2009 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26 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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	Introduction	8
36	1 Scope	9
37	1.1 In-Scope.....	9
38	1.2 Out of Scope	9
39	1.3 Conformance	9
40	2 Normative References.....	9
41	2.1 Approved References	9
42	3 Terms and Definitions.....	10
43	4 Symbols and Abbreviated Terms.....	11
44	5 Prefixes and XML Namespaces	12
45	6 WS-Management Default Addressing Model	13
46	6.1 Class-Specific ResourceURI.....	13
47	6.2 “All Classes” ResourceURI	14
48	6.3 Accounting for Different CIM Namespaces.....	15
49	7 Accessing Instances.....	16
50	7.1 Get	16
51	7.2 Put.....	16
52	7.3 Delete.....	16
53	7.4 Create	17
54	8 Filter Dialects.....	17
55	8.1 CQL.....	17
56	8.2 Association Queries	19
57	9 Enumeration	25
58	9.1 EnumerationMode.....	25
59	9.2 XmlFragment	26
60	9.3 Polymorphism	27
61	9.4 XPath Enumeration Using the Class-Specific ResourceURI	29
62	9.5 XPath Enumerate Using the “All Classes” ResourceURI	30
63	10 Subscriptions.....	30
64	10.1 Indication Filters.....	31
65	10.2 Subscribe Request.....	32
66	10.3 Subscription Response.....	36
67	10.4 Event Delivery.....	36
68	10.5 Subscription Reporting.....	37
69	10.6 Unsubscribe and Renew Requests	40
70	11 Extrinsic Methods	41
71	12 Exceptions	41
72	13 CIM Specific WS-Management Options.....	42
73	13.1 ShowExtensions Option.....	42
74	14 Instance Representation	43
75	15 Fault Codes	44
76	15.1 wsmb:CIMException	44
77	15.2 wsmb:PolymorphismModeNotSupported	44
78	16 Mapping for DSP0200 CIM Operations.....	45
79	16.1 Supported Operations.....	45
80	16.2 Unsupported Operations.....	67

81 17 Mapping of Error Messages to SOAP Fault Subcodes 68
 82 18 XSD 69
 83 19 WSDL 69
 84

85 **Tables**

86 Table 1 – Prefixes and XML Namespaces 12
 87 Table 2 – CIM_IndicationFilter Properties 38
 88 Table 3 – CIM_ListenerDestinationWSManagement Required Properties 38
 89 Table 4 – CIM_ListenerDestinationWSManagement Optional Properties 38
 90 Table 5 – Required Properties for CIM_IndicationSubscription and CIM_FilterCollectionSubscription..... 39
 91 Table 6 – wsmb:CIMException 44
 92 Table 7 – wsmb:PolymorphismModeNotSupported 44
 93 Table 8 – GetInstance 46
 94 Table 9 – GetInstance Arguments 46
 95 Table 10 – GetInstance Error Codes 47
 96 Table 11 – DeleteInstance 47
 97 Table 12 – DeleteInstance Arguments 47
 98 Table 13 – DeleteInstance Error Codes 47
 99 Table 14 – ModifyInstance 48
 100 Table 15 – ModifyInstance Arguments 48
 101 Table 16 – ModifyInstance Error Codes 48
 102 Table 17 – CreateInstance 49
 103 Table 18 – CreateInstance Arguments 49
 104 Table 19 – CreateInstance Error Codes 49
 105 Table 20 – EnumerateInstances 50
 106 Table 21 – EnumerateInstances Arguments 50
 107 Table 22 – EnumerateInstances Error Codes 51
 108 Table 23 – EnumerateInstanceNames 51
 109 Table 24 – EnumerateInstanceNames Arguments 51
 110 Table 25 – EnumerateInstanceNames Error Codes 52
 111 Table 26 – Associators 52
 112 Table 27 – Associators Arguments 52
 113 Table 28 – Associators Error Codes 53
 114 Table 29 – AssociatorNames 53
 115 Table 30 – AssociatorNames Arguments 54
 116 Table 31 – AssociatorNames Error Codes 54
 117 Table 32 – References 54
 118 Table 33 – References Arguments 55
 119 Table 34 – References Error Codes 55
 120 Table 35 – ReferenceNames 55
 121 Table 36 – ReferenceNames Arguments 56
 122 Table 37 – ReferenceNames Error Codes 56
 123 Table 38 – OpenEnumerateInstances 57
 124 Table 39 – OpenEnumerateInstances Arguments 57
 125 Table 40 – OpenEnumerateInstances Error Codes 57

126 Table 41 – OpenEnumerateInstancePaths..... 58

127 Table 42 – OpenEnumerateInstancePaths Arguments 58

128 Table 43 – OpenEnumerateInstancePaths Error Codes 59

129 Table 44 – OpenReferenceInstances 59

130 Table 45 – OpenReferenceInstances Arguments..... 59

131 Table 46 – OpenReferenceInstances Error Codes..... 60

132 Table 47 – OpenReferenceInstancePaths..... 60

133 Table 48 – OpenReferenceInstancePaths Arguments 61

134 Table 49 – OpenReferenceInstancePaths Error Codes 61

135 Table 50 – OpenAssociatorInstances 62

136 Table 51 – OpenAssociatorInstances Arguments 62

137 Table 52 – OpenAssociatorInstances Error Codes 63

138 Table 53 – OpenAssociatorInstancePaths..... 63

139 Table 54 – OpenAssociatorInstancePaths Arguments 63

140 Table 55 – OpenAssociatorInstancePaths Error Codes 64

141 Table 56 – PullInstancesWithPath 64

142 Table 57 – PullInstancesWithPath Arguments..... 65

143 Table 58 – PullInstancesWithPath Error Codes..... 65

144 Table 59 – PullInstancePaths 65

145 Table 60 – PullInstancePaths Arguments..... 66

146 Table 61 – PullInstancePaths Error Codes..... 66

147 Table 62 – CloseEnumeration..... 66

148 Table 63 – CloseEnumeration Arguments 66

149 Table 64 – CloseEnumeration Error Codes 67

150 Table 65 – CIM Error Messages with Corresponding Subcode Mappings..... 68

151

153

Foreword

154 The *WS-Management CIM Binding Specification* (DSP0227) was prepared by the DMTF WS-
155 Management Working Group.

156 **Acknowledgments**

157 The authors wish to acknowledge the following people:

158 **Editors:**

- 159 • Richard Landau – Dell Inc.
- 160 • Hemal Shah – Broadcom Corporation
- 161 • Steve Hand – Symantec Corp.

162 **Contributors:**

- 163 • Josh Cohen – Microsoft Corporation (Chair)
- 164 • Jim Davis – WBEM Solutions
- 165 • David Hines – Intel
- 166 • Bryan Murray – Hewlett-Packard
- 167 • Brian Reistad – Microsoft Corporation

168

169

Introduction

170 This document describes the CIM binding for WS-Management. It describes how transformed CIM
171 resources, as specified by the [WS-CIM Mapping Specification](#), are bound to WS-Management operations
172 and WSDL definitions.

173

WS-Management CIM Binding Specification

174 1 Scope

175 This clause describes the scope of this specification, including some items that are specifically out of
176 scope.

177 1.1 In-Scope

178 This specification describes how to use the Web Services for Management (WS-Management) protocol to
179 communicate with resources modeled with CIM and exposed through the XML schema mapping described
180 by WS-CIM.

181 1.2 Out of Scope

182 This specification does not describe how to expose the WBEM intrinsic methods that perform schema
183 manipulation of CIM classes (for example, CreateClass) using the WS-Management protocol.

184 This specification does not describe how to generate the XML schema for a CIM class.

185 1.3 Conformance

186 This specification supplements the [WS-Management Specification](#). When this specification is supported,
187 requests using a particular version of WS-Management are assumed to use the same version of this
188 specification; both specifications will be updated concurrently. (The version of this specification cannot
189 generally be directly determined from a SOAP message because most requests do not contain any
190 elements from this specification or the XML namespace of this specification.)

191 An implementation is not conformant with this specification if it fails to satisfy one or more of the
192 requirements defined in the conformance rules for each clause, as indicated by the following format:

193 **Rnnnn:** Rule text

194 2 Normative References

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

198 2.1 Approved References

199 DMTF DSP0004, *CIM Infrastructure Specification 2.3*,
200 http://www.dmtf.org/standards/published_documents/DSP0004_2.3.pdf

201 DMTF DSP0200, *Specification for CIM Operations over HTTP 1.3*,
202 http://www.dmtf.org/standards/published_documents/DSP0200_1.3.pdf

203 DMTF DSP0226, *WS-Management Specification, 1.0*,
204 http://www.dmtf.org/standards/published_documents/DSP0226_1.0.pdf

205 DMTF DSP0230, *WS-CIM Mapping Specification, 1.0*,
206 http://www.dmtf.org/standards/published_documents/DSP0230_1.0.pdf

- 207 DMTF DSP8016, *WBEM Operations Message Registry 1.0*,
208 http://schemas.dmtf.org/wbem/messageregistry/1/DSP8016_1.0.xml
- 209 IETF RFC3986, *Uniform Resource Identifier (URI) Generic Syntax, January 2005*,
210 <http://www.ietf.org/rfc/rfc3986.txt>
- 211 IETF RFC4646, *Tags for Identifying Languages, September 2006*, <http://www.ietf.org/rfc/rfc4646.txt>
- 212 W3C, *Namespaces in XML, W3C Recommendations, 14 January 1999*,
213 <http://www.w3.org/TR/1999/REC-xml-names-19990114>
- 214 W3C, *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition) SOAP, 1.2 , W3C*
215 *Recommendation, 27 April 2007*, <http://www.w3.org/TR/soap12-part1/>
- 216 W3C, *Web Services Description Language (WSDL), 1.1, W3C Note, 15 March 2001*,
217 <http://www.w3.org/TR/wsdl>
- 218 W3C, *Web Services Addressing (WS-Addressing), W3C Member Submission, 10 August 2004*,
219 <http://www.w3.org/Submission/ws-addressing/>
- 220 W3C, *Web Services Enumeration (WS-Enumeration), W3C Member Submission, 15 March 2006*,
221 <http://www.w3.org/Submission/WS-Enumeration/>
- 222 W3C, *Web Services Eventing (WS-Eventing), W3C Member Submission 15 March 2006*,
223 <http://www.w3.org/Submission/WS-Eventing/>
- 224 W3C, *Web Services Transfer (WS-Transfer), W3C Member Submission, 27 September 2006*,
225 <http://www.w3.org/Submission/WS-Transfer/>
- 226 W3C, *XML Path Language (XPath) Version 1.0, W3C Recommendation, 16 November 1999*,
227 <http://www.w3.org/TR/1999/REC-xpath-19991116>
- 228 W3C, *XML Schema Part 1: Structures Second Edition, W3C Recommendation, 28 October 2004*,
229 <http://www.w3.org/TR/xmlschema-1/>
- 230 W3C, *XML Schema Part 2: Datatypes Second Edition, W3C Recommendation, 28 October 2004*,
231 <http://www.w3.org/TR/xmlschema-2/>

232 **3 Terms and Definitions**

233 The terms used in [DSP0226](#) and [DSP0230](#) also apply to this specification.

234 **3.1**

235 **can**

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

237 **3.2**

238 **cannot**

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

240 **3.3**

241 **conditional**

242 indicates requirements to be followed strictly in order to conform to the document when the specified
243 conditions are met

- 244 **3.4**
245 **mandatory**
246 indicates requirements to be followed strictly in order to conform to the document and from which no
247 deviation is permitted
- 248 **3.5**
249 **may**
250 indicates a course of action permissible within the limits of the document
- 251 **3.6**
252 **need not**
253 indicates a course of action permissible within the limits of the document
- 254 **3.7**
255 **optional**
256 indicates a course of action permissible within the limits of the document
- 257 **3.8**
258 **shall**
259 indicates requirements to be followed strictly in order to conform to the document and from which no
260 deviation is permitted
- 261 **3.9**
262 **shall not**
263 indicates requirements to be followed strictly in order to conform to the document and from which no
264 deviation is permitted
- 265 **3.10**
266 **should**
267 indicates that among several possibilities, one is recommended as particularly suitable, without mentioning
268 or excluding others, or that a certain course of action is preferred but not necessarily required
- 269 **3.11**
270 **should not**
271 indicates that a certain possibility or course of action is deprecated but not prohibited
- 272 **3.12**
273 **unspecified**
274 indicates that this profile does not define any constraints for the referenced CIM element or operation
- 275 **3.13**
276 **base class**
277 A class that is defined in a CIM schema and from which other classes are derived which may contain other
278 properties or other CIM named elements. These additional named elements are extensions to the base
279 class.
- 280 **4 Symbols and Abbreviated Terms**
- 281 **4.1**
282 **CQL**
283 CIM Query Language

- 284 **4.2**
- 285 **EPR**
- 286 Endpoint Reference
- 287 **4.3**
- 288 **GED**
- 289 Global Element Definition
- 290 **4.4**
- 291 **URI**
- 292 Uniform Resource Identifier
- 293 **4.5**
- 294 **WBEM**
- 295 Web-Based Enterprise Management
- 296 **4.6**
- 297 **WSDL**
- 298 Web Services Description Language
- 299 **4.7**
- 300 **XSD**
- 301 XML Schema Definition

302 **5 Prefixes and XML Namespaces**

303 Table 1 lists namespaces that are used in this specification. The choice of any namespace prefix is
 304 arbitrary and not semantically significant.

305 **Table 1 – Prefixes and XML Namespaces**

Prefix	XML Namespace	Reference
wsmb	http://schemas.dmtf.org/wbem/wsman/1/cimbinding.xsd	This specification
wsman	http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd	WS-Management
cim	http://schemas.dmtf.org/wbem/wscim/1/common	WS-CIM
s	http://www.w3.org/2003/05/soap-envelope	SOAP 1.2
xs	http://www.w3.org/2001/XMLSchema	XML Schema
wsdl	http://schemas.xmlsoap.org/wsdl	WSDL 1.1
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing	WS-Addressing
wsen	http://schemas.xmlsoap.org/ws/2004/09/enumeration	WS-Enumeration
wxf	http://schemas.xmlsoap.org/ws/2004/09/transfer	WS-Transfer
wse	http://schemas.xmlsoap.org/ws/2004/08/eventing	WS-Eventing

306 6 WS-Management Default Addressing Model

307 WS-Management defines a default addressing model based on WS-Addressing. This clause describes how
308 CIM objects are addressed when they are accessed with the protocol.

309 WS-Management makes use of WS-Addressing to identify and access resources. WS-Management
310 defines a reference format using the WS-Addressing EndpointReference element, making use of the
311 ReferenceParameter field to contain specific elements (ResourceURI and SelectorSet) to aid in identifying
312 the desired object or objects.

313 **R6-1:** Services that support the default addressing model defined by WS-Management are required
314 to conform to this clause and its subclauses.

315 6.1 Class-Specific ResourceURI

316 For standard CIM classes, the ResourceURI is identical to the XML namespace URI of the schema for the
317 class. This ResourceURI targets the named class and any derived classes depending on the role of
318 polymorphism.

319 **R6.1-1:** Instances of a specific class shall be addressed using a ResourceURI that identifies a specific
320 class.

321 EXAMPLE: The following ResourceURI is used to reference the CIM_SoftwareElement class in version 2 of the CIM
322 schema.

```
323 (01) http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_SoftwareElement
```

324 Note that the XML schema namespace for the instances never changes to reflect CIM namespace usage;
325 only the ResourceURI changes. Class definitions are pure schema; they are independent of their scope or
326 CIM namespace residence. See 6.3 for a description of classes that reside in explicit namespaces.

327 **R6.1-2:** It is recommended that vendor-defined classes use the same value for ResourceURI that is
328 used for the XML namespace of the class. The vendor-defined XML namespace should include some
329 form of version field in the namespace URI that can be changed when backward-incompatible changes
330 are made to the XML schema.

331 Resources without keys are referenced by a class-specific ResourceURI within the SOAP binding, as
332 follows:

```
333 (1) <s:Envelope ...>  
334 (2)   <s:Header>  
335 (3)     <wsa:To> network address </wsa:To>  
336 (4)     <wsman:ResourceURI> URI of the item </wsman:ResourceURI>  
337 (5)   </s:Header>
```

338 **R6.1-3:** If keys are required to discriminate among instances, the WS-Management SelectorSet SOAP
339 header shall be used, as follows:

```
340 (6) <s:Envelope ...>  
341 (7)   <s:Header>  
342 (8)     <wsa:To> network address </wsa:To>  
343 (9)     <wsman:ResourceURI> URI of the item </wsman:ResourceURI>  
344 (10)    <wsman:SelectorSet>  
345 (11)      <wsman:Selector Name="KeyName"> Key Value </wsman:Selector>  
346 (12)    </wsman:SelectorSet>  
347 (13)    ...  
348 (14)  </s:Header>
```

349 In this case, the key values required by CIM become individual Selector values. The name of the key is
 350 repeated in the Name attribute, and the key value becomes the value of the Selector element. Note that all
 351 CIM instances except indications have keys.

352 EXAMPLE: Example class definition:

```

353 (15) class CIM_SoftwareElement : CIM_LogicalElement
354 (16) {
355 (17)     [key] string Name;
356 (18)     [key] string Version;
357 (19)     [key] uint16 SoftwareElementState;
358 (20)     [key] string SoftwareElementID;
359 (21)     [key] uint16 TargetOperatingSystem;
360 (22)     string OtherTargetOS;
361 (23)     string Manufacturer;
362 (24)     string BuildNumber;
363 (25)     string SerialNumber;
364 (26)     string CodeSet;
365 (27)     string IdentificationCode;
366 (28)     string LanguageEdition;
367 (29) };
  
```

368 **R6.1-4:** The ResourceURI shall be the XML namespace for the class, and the zero or more Selectors
 369 shall contain keys defined by this class. A service may process a request with a subset of the keys if
 370 the subset uniquely identifies the instance. Clients are guaranteed correct behavior if they supply all
 371 keys in the request. Clients might encounter different behavior at different resources if they do not
 372 supply all keys.

373 EXAMPLE: The following example illustrates how to form a EPR using the class definition above:

```

374 (1) <s:Header>
375 (2)     <wsa:To> network address </wsa:To>
376 (3)     <wsman:ResourceURI>
377 (4)         http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_SoftwareElement
378 (5)     </wsman:ResourceURI>
379 (6)     <wsman:SelectorSet>
380 (7)         <wsman:Selector Name="Name"> AcmeCAD </wsman:Selector>
381 (8)         <wsman:Selector Name="Version"> 1.2 </wsman:Selector>
382 (9)         <wsman:Selector Name="SoftwareElementState"> 1 </wsman:Selector>
383 (10)        <wsman:Selector Name="SoftwareElementID"> 123F00 </ wsman:Selector>
384 (11)        <wsman:Selector Name="TargetOperatingSystem"> 12 </wsman:Selector>
385 (12)     </wsman:SelectorSet>
386 (13)     ...
387 (14) </s:Header>
  
```

388 **R6.1-5:** A service shall accept a properly-formed endpoint reference that specifies a class-specific
 389 ResourceURI and keys, if necessary, as defined in this clause.

390 6.2 “All Classes” ResourceURI

391 Because certain types of queries may cross class boundaries, the class-specific ResourceURI defined in
 392 6.1 is not always applicable.

393 **R6.2-1:** Services supporting cross-class queries shall accept an “all classes” ResourceURI.

394 This ResourceURI effectively targets the query processor in the CIM Server itself and can be used to return
 395 both CIM and vendor classes.

396 The "all classes" ResourceURI is of the same form as the class-specific ResourceURI in which the schema
 397 version and class name are replaced with the star character. The presence of the WS-CIM version in this
 398 ResourceURI allows the client to indicate which version of the [WS-CIM Mapping Specification](#) should be
 399 used in the translation of the CIM instances to XML.

400 For example, the following ResourceURI refers to all classes in the CIM namespace represented using
 401 version 1 of WS-CIM:

```
402 http://schemas.dmtf.org/wbem/wscim/1/*
```

403 When using the class-specific ResourceURI, the results of the enumeration may contain instances of the
 404 class identified in the ResourceURI or any derived class. However, the class name is typically repeated in
 405 both the ResourceURI and the filter expression.

406 The advantage to the "all classes" construct is that a single URI may be used for all resource queries and
 407 the class information appears in only one place: the filter expression. When the "all classes" construct is
 408 used in an Enumerate request, the results returned contain instances from a single CIM namespace, with
 409 one important exception: a query using an associationFilter filter dialect such as AssociatedInstances may
 410 return instances from more than one CIM namespace.

411 6.3 Accounting for Different CIM Namespaces

412 The following special Selector Name is defined to indicate the CIM namespace of the resource or
 413 resources for which the message is intended:

```
414 <wsman:Selector Name="__cimnamespace">xs:string</wsman:Selector>
```

415 This selector is in addition to any other selectors for CIM keys and is unlikely to collide with others because
 416 most CIM keys do not start with two underscore (__) characters.

417 The absence of this Selector Name in a message indicates that the intended resources are in the default
 418 CIM namespace for that service. This specification does not define what the default CIM namespace
 419 should be.

420 **R6.3-1:** A service offering more than one CIM namespace should include the __cimnamespace
 421 Selector Name in an EPR returned in a response message to identify the CIM namespace of an
 422 instance in the response. New implementations are strongly encouraged to include the
 423 __cimnamespace selector; alternate methods of conveying the CIM Namespace may be deprecated in
 424 the future.

425 **R6.3-2:** A service shall not fault if the __cimnamespace Selector Name is absent and instead shall
 426 utilize the default CIM namespace.

427 **R6.3-3:** A service offering more than one CIM namespace should indicate in metadata which CIM
 428 namespace is the default. This specification does not define the location or format of such metadata.

429 **R6.3-4:** A service supporting more than one CIM namespace shall fault a request that specifies a
 430 namespace whose name is not one of the names of the CIM namespaces supported.

431 **R6.3-5:** If a service supports exactly one namespace, then

- 432 a) the service shall fault a request that includes a __cimnamespace selector that does not
- 433 match the name of the single namespace; and
- 434 b) the service should include the __cimnamespace selector in an EPR returned in a response
- 435 message to identify the CIM namespace of an instance in the response.

436 In all cases, R6.3-2 applies: a request with no __cimnamespace selector utilizes the default
 437 namespace. If a service supports only one namespace, then that namespace is the default.

438 7 Accessing Instances

439 When retrieving and updating an instance of a class, the WS-Transfer Get, Put, and Delete operations from
440 WS-Transfer are used. When creating an instance of a class, the WS-Transfer create operation is used.
441 The fragment access SOAP header defined by WS-Management may be applied to these operations.

442 Class inheritance also affects how WS-Transfer operations are specified. In many cases vendors have
443 derived a vendor-specific class from the CIM class that allows multiple vendors to implement the same
444 class in the same CIM namespace even if they have not added any additional properties. For example, an
445 implementation may choose to instantiate Vendor_ComputerSystem, which is derived from
446 CIM_ComputerSystem. In many cases, a client must access instances of the derived class, but has only
447 the name of the base class. To access an instance of such a derived class, or obtain an EPR for such an
448 instance that can be used in WS-Transfer operations, a client generally will enumerate instances using the
449 base class. The returned instances or EPRs can optionally contain the correct derived classname. See
450 section 9.3 (“Polymorphism”) for details.

451 The XML Schema representation of CIM instances permits the omission of non-key and non-required
452 properties in their corresponding XML instance documents. The [WS-CIM Mapping Specification](#) (DSP0230)
453 defines runtime rules for the Get, Delete, and Create operations.

454 **R7-1:** A service should return a wsa:ActionNotSupported fault if the “all classes” ResourceURI is
455 used with any of the WS-Transfer operations, even if this ResourceURI is supported for enumerations
456 or eventing.

457 7.1 Get

458 The following clause defines requirements and presents examples related to getting instances.

459 **R7.1-1:** A service supporting the Get operation and using the WS-Management Default Addressing
460 Model shall support access using the class-specific ResourceURI that corresponds to the creation
461 class and the selectors of the given instance.

462 **R7.1-2:** The response representation shall use the XML Schema identified by the class in the
463 ResourceURI.

464 7.2 Put

465 The following clause defines requirements and presents examples related to putting or modifying
466 instances.

467 **R7.2-1:** A service supporting the Put operation and using the WS-Management Default Addressing
468 Model shall support access using the class-specific ResourceURI that corresponds to the creation
469 class and the selectors of the given instance.

470 **R7.2-2:** A service supporting the Put operation shall accept instance representations that have omitted
471 schema-optional. Any elements not included in the transfer operation shall be left unchanged. A
472 service supporting fragment-level put operations shall also observe this behavior.

473 **R7.2-3:** The request and response representations shall use the XML Schema identified by the class in
474 the ResourceURI.

475 7.3 Delete

476 The following clause defines requirements and presents examples related to deleting instances.

477 **R7.3-1:** A service supporting the Delete operation and using the WS-Management Default Addressing
478 Model shall support access using the class-specific ResourceURI that corresponds to the creation
479 class and the selectors of the given instance.

480 7.4 Create

481 The Create operation is different from the other WS-Transfer operations because it is sent to a resource
482 factory rather than to a resource. For CIM, the class-specific ResourceURI is the factory resource that can
483 be used to create instances of the class.

484 **R7.4-1:** A service supporting the Create operation and using the WS-Management Default Addressing
485 Model shall support access using the class-specific ResourceURI corresponding to the creation class
486 and, if warranted, the __cimnamespace Selector Name.

487 However, the fragment-level Create operation operates on the resource itself, so it behaves in the same
488 fashion as the Put operation:

489 **R7.4-2:** A service may support the fragment-level Create operation using the class-specific
490 ResourceURI that corresponds to the creation class and the selectors of the given instance.

491 **R7.4-3:** A service supporting the Create operation shall accept instance representations that have
492 omitted schema-optional properties and shall interpret such omissions as a request to create the object
493 with the corresponding omitted properties set to a value of NULL. A service supporting the fragment-
494 level Create operation shall also observe this behavior.

495 8 Filter Dialects

496 Both [WS-Enumeration](#) and [WS-Eventing](#) define XPath Version 1.0 as the default filter language (called a
497 "dialect" in those specifications), though other filter languages are accommodated. This specification
498 defines two additional dialects for use with resources modeled using CIM. Services may support these and
499 other query languages by accepting messages with appropriate dialect URIs.

500 The filter dialects defined in this clause are intended for use with WS-Enumeration and WS-Eventing and
501 not with Fragment-level WS-Transfer.

502 8.1 CQL

503 CQL is a SQL-based query language that includes the class name as part of the query. The dialect filter
504 URI for this language is as follows:

505 <http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf>

506 **R8.1-1:** Services that accept CQL statements of the form "select * from ..." shall return each instance
507 representation using the GED defined for the object's class within the wsen:Items element.

508 **R8.1-2:** Services that accept CQL statements of the form "select a,b,c from ..." (a query with projection)
509 shall return each instance representation using the wsman:XmlFragment element. Within the
510 wsman:XmlFragment element, the service shall return property values named in the select statement
511 using either an element with the given label if the AS keyword is used or the property's GED defined in
512 the [WS-CIM Mapping Specification](#) if the select-list entry is a property (ignoring any chain or property-
513 scope). Expressions and literals without AS keywords are not valid CQL expressions.

514 Clients should use wsman:Filter, as opposed to wsen:Filter or wse:Filter, when using CQL statements of the
515 form "select a,b,c from ..." because these queries contain projections and are not Boolean predicates.

516 **R8.1-3:** Services supporting CQL statements of the form "select a,b,c from ..." may return results in any
517 order. To provide clients a mechanism to correlate results with the CQL expression, services should
518 include the attribute wsmb:Expression for all selected-entry elements, and shall include the attribute
519 wsmb:Expression for any selected-entry that would have a duplicate name with another selected-entry.
520 The value of the wsmb:Expression attribute on the element shall be the selected-entry in the select-list
521 from which the element resulted.

522 EXAMPLE 1: If the select-list of a CQL statement is "ID, Foo.Name, Bar::Host, A AS B, X * Y AS Z", the query
 523 returns the associated elements in the following fragment:

```
524 (1) <wsen:Items xmlns:ex='...'>
525 (2)   <wsman:XmlFragment>
526 (3)     <ex:ID>...</ex:ID>
527 (4)     <ex:Name>...</ex:Name>
528 (5)     <ex:Host>...</ex:Host>
529 (6)     <B>...</B>
530 (7)     <Z>...</Z>
531 (8)   </wsman:XmlFragment>
532 (9) </wsen:Items>
```

533 NOTE 1: The elements that result from the AS keyword do not have an XML namespace.

534 NOTE 2: Because the response elements are wrapped in the XmlFragment element, which is defined to turn off
 535 validation for the entire content of the XmlFragment, it is permissible for the service not to include namespace prefixes
 536 for the enclosed elements.

537 If a join were used with the same named property included from both classes, then the wsmb:Expression
 538 would be used to differentiate between them.

539 EXAMPLE 2: Given a select-list of "CIM_Foo.ID, CIM_Foo.Name, CIM_Bar.Name" the associated elements would
 540 be as follows:

```
541 (1) <wsen:Items xmlns:bar='...' xmlns:foo='...'>
542 (2)   <wsman:XmlFragment>
543 (3)     <foo:ID>...</foo:ID>
544 (4)     <bar:Name wsmb:Expression='CIM_Bar.Name'> ...</bar:Name>
545 (5)     <foo:Name wsmb:Expression='CIM_Foo.Name'> ...</foo:Name>
546 (6)   </wsman:XmlFragment>
547 (7) </wsen:Items>
```

548 **R8.1-4:** If a service supports wsman:EnumerationMode=EnumerateObjectAndEPR for enumerating
 549 instances and endpoint references, then it shall compose the instance representation of the results of
 550 the CQL query (as specified in the previous two rules) with the EPR. The CQL query selects the
 551 instances and properties of the instance to be returned but has no effect on the EPR that refers to
 552 objects that match the where clause of the CQL query.

553 **R8.1-5:** If a service supports wsman:EnumerationMode=EnumerateEPR for enumerating endpoint
 554 references, then it shall return the EPRs for instances that match the where clause of the CQL query
 555 and ignore any properties specified in the select portion of the CQL query.

556 **R8.1-6:** If a service uses the WS-Management Default Addressing model, then it should support this
 557 filter dialect for Enumerate operations. If the CQL dialect is not supported by the addressed endpoint
 558 service, the service shall respond with a wsen:FilterDialectRequestedUnavailable fault.

559 **R8.1-7:** If a service uses the WS-Management Default Addressing model and supports the CQL dialect
 560 for Enumerate operations it shall support addressing the CIM Server (through the "all classes"
 561 ResourceURI) and it should support addressing instances of a class (through the class-specific
 562 ResourceURI). If the CQL query references in the FROM clause more than one CIM class, then the
 563 Enumerate operation shall be addressed to the "all classes" ResourceURI. If the addressed endpoint
 564 and the query contradict each other (for example, the CIM classname in the class-specific
 565 ResourceURI does not match the CIM classname in the CQL FROM clause), the service shall respond
 566 with a wsen:CannotProcessFilter fault.

567 **R8.1-8:** If a service uses the WS-Management Default Addressing model it should support this filter
 568 dialect for Subscribe operations. If the CQL dialect is not supported by the addressed endpoint service,
 569 the service shall respond with a wsen:FilterDialectRequestedUnavailable fault.

570 **R8.1-9:** If a service uses the WS-Management Default Addressing model and supports the CQL dialect
 571 for Subscribe operations it shall support addressing the CIM Server (through the “all classes”
 572 ResourceURI) and it should support addressing instances of a class (through the class-specific
 573 ResourceURI). If the addressed endpoint and the query contradict each other (for example, the CIM
 574 classname in the class-specific ResourceURI does not match the CIM classname in the CQL FROM
 575 clause), the service shall respond with a wse:EventSourceUnableToProcess fault.

576 **R8.1-10:** Services that accept CQL queries should return instances of the most-derived class rather
 577 than a requested class, even though the query names a specific class.

578 **EXAMPLE 3:** The following request issues a CQL query in which the returned results include properties from the
 579 selected instances. This example uses the WS-Management Default Addressing Model but applies to
 580 any EPR model used by the service.

```

581 (1) <s:Envelope>
582 (2)   <s:Header>
583 (3)     <wsman:ResourceURI>
584 (4)       http://schemas.dmtf.org/wbem/wscim/1/*
585 (5)     </wsman:ResourceURI>
586 (6)   </s:Header>
587 (7)   <s:Body>
588 (8)     <wsen:Enumerate>
589 (9)       <wsman:Filter Dialect="http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf">
590 (10)        SELECT Name, PrimaryOwnerName
591 (11)        FROM CIM_ComputerSystem
592 (12)        WHERE EnabledState = 3
593 (13)       </wsman:Filter>
594 (14)     </wsen:Enumerate>
595 (15)   </s:Body>
596 (16) </s:Envelope>

```

597 The results include the two requested properties for instances that are “Disabled”:

```

598 (1) <s:Body>
599 (2)   <wsen:PullResponse>
600 (3)     <wsen:EnumerationContext> ... </wsen:EnumerationContext>
601 (4)     <wsen:Items>
602 (5)       <wsman:XmlFragment>
603 (6)         <Name>system1</Name>
604 (7)         <PrimaryOwnerName>Joe</PrimaryOwnerName>
605 (8)       </wsman:XmlFragment>
606 (9)       <wsman:XmlFragment>
607 (10)        <Name>system2</Name>
608 (11)        <PrimaryOwnerName>Mary</PrimaryOwnerName>
609 (12)       </wsman:XmlFragment>
610 (13)       ... etc.
611 (14)     </wsen:Items>
612 (15)   </wsen:PullResponse>
613 (16) </s:Body>

```

614 8.2 Association Queries

615 CIM uses associations to relate instances of different classes and defines intrinsic operations to find related
 616 classes. Association queries start with one instance that participates in the association (called the source
 617 object) and finds all related instances (called the result objects) linked through associations in which a

618 reference to the source object appears as the value of a specific property (called the role) in the
 619 association. The query can be further constrained by limiting the roles that are used for the source or result
 620 objects as well as limiting the type of the association and result classes. Alternatively, it is possible to issue
 621 a query for instances of the associations themselves using a similar set of constraining parameters.

622 This specification defines the following dialect filter URI for association queries:

623 `http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter`

624 The following rules apply only to services that support association queries:

625 **R8.2-1:** If a service uses the WS-Management Default Addressing model it should support the
 626 association filter dialect for Enumerate operations that are addressed to the “all classes” ResourceURI.
 627 If such a service receives an Enumerate request addressed to a class-specific Resource URI
 628 specifying this filter dialect, the service shall respond with a wsen:FilterDialectRequestedUnavailable
 629 fault.

630 **R8.2-2:** If a service supports wsman:EnumerationMode=EnumerateObjectAndEPR for enumerating
 631 endpoint references, then it shall compose the instance representation of the results of the association
 632 query with the EPR as directed. The association query selects the instances and properties of the
 633 instance to be returned but has no effect on the presence or absence of the EPR.

634 **R8.2-3:** The service should return a wse:FilteringRequestedUnavailable fault in response to Subscribe
 635 requests using the association filter dialect.

636 **R8.2-4:** If the result of an association query includes no instances, the service shall not return a fault.

637 8.2.1 Associated Instances

638 For queries that return associated instances, the Enumerate message has the following form:

```

639 (1) <wsen:Enumerate>
640 (2)   <wsman:Filter
641 (3)     Dialect="http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter">
642 (4)     <wsmb:AssociatedInstances>
643 (5)       <wsmb:Object> xs:any </wsmb:Object>
644 (6)       <wsmb:AssociationClassName> xs:NCName </wsmb:AssociationClassName> ?
645 (7)       <wsmb:Role> xs:NCName </wsmb:Role> ?
646 (8)       <wsmb:ResultClassName> xs:NCName </wsmb:ResultClassName> ?
647 (9)       <wsmb:ResultRole> xs:NCName </wsmb:ResultRole> ?
648 (10)      <wsmb:IncludeResultProperty> xs:NCName </wsmb:IncludeResultProperty> *
649 (11)     </wsmb:AssociatedInstances>
650 (12)   </wsman:Filter>
651 (13) </wsen:Enumerate>
  
```

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

- 653 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances

654 The results include instances related to the source object through an association.

655 **R8.2.1-1:** The results of the enumeration shall be instances associated with the object through an
 656 association instance subject to the additional constraints listed in this clause.

- 657 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances/wsmb:Object

658 Identifies the source object for the association query and is required.

- 659 **R8.2.1-2:** The results shall be associated with the object identified by the endpoint reference in
660 wsmb:Object.
- 661 **R8.2.1-3:** If the EPR to which the Enumerate message is sent and the EPR of the source object
662 reference two different CIM namespaces, the service may respond with a wsen:CannotProcessFilter
663 fault.
- 664 **R8.2.1-4:** If the EPR of the source object does not reference exactly one valid CIM instance, the
665 service shall respond with a wsen:CannotProcessFilter fault. Services should include a textual
666 description of the problem.
- 667 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances/wsmb:AssociationClassName
668 Represents the name of a CIM association class. This element or parameter is optional.
- 669 **R8.2.1-5:** If the AssociationClassName is present, the results shall include only the instances related to
670 the source object through associations that are instances of only the named class or derived classes. If
671 the AssociationClassName is absent, results shall include instances that are related to the source
672 object through associations of any type.
- 673 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances/wsmb:Role
674 Represents the name of a reference property of a CIM association class. This element or parameter is
675 optional.
- 676 **R8.2.1-6:** If the Role name is present, the results shall include only instances related to the source
677 object through an association in which the source object plays the specified role (that is, the name of
678 the property in the association class that refers to the source object shall match the value of this
679 parameter). If the Role name is absent, the results shall include instances associated to the source
680 regardless of the role of the source object in the association.
- 681 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances/wsmb:ResultClassName
682 Represents the name of a CIM class. This element or parameters is optional.
- 683 **R8.2.1-7:** If the ResultClassName is present, the results shall include only objects that are instances of
684 the named class or any of its derived classes. If the ResultClassName is absent, the results shall
685 include all objects regardless of type.
- 686 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances/wsmb:ResultRole
687 Represents the name of a reference property of a CIM association class. This element or parameter is
688 optional.
- 689 **R8.2.1-8:** If ResultRole name is present, the results shall only include instances related to the source
690 object via an association in which the returned object plays the specified role. In other words, the name
691 of the property in the association class that refers to the returned object shall match the value of this
692 parameter.
- 693 • wsen:Enumerate/wsman:Filter/wsmb:AssociatedInstances/wsmb:IncludeResultProperty
694 Represents the name of one or more properties of a CIM class. This element or parameter is optional.
- 695 **R8.2.1-9:** If the query does not include an IncludeResultProperty element, the service shall return each
696 instance representation using the GED defined for the object's class within the wsen:Items element.
- 697 **R8.2.1-10:** If the query includes one or more IncludeResultProperty elements, the service shall
698 return each instance representation using the wsman:XmlFragment element. Within the
699 wsman:XmlFragment element, the service shall return property values using the property GEDs
700 defined in the [WS-CIM Mapping Specification](#). If the query includes one or more IncludeResultProperty
701 elements, the service shall not return any IncludeResultProperty elements not specified. The service
702 shall ignore any IncludeResultProperty elements that describe properties not defined by the target

703 class. If the service does not support fragment-level access, it shall return a
704 wsman:UnsupportedFeature fault with the following detail code:

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

706 **R8.2.1-11:** A service may omit returned properties, even when explicitly requested, if and only if
707 such properties have not been set (that is, the properties have a NULL value). The requestor is to
708 interpret the absence of these properties as the properties having a NULL value.

709 **R8.2.1-12:** A service shall not return a fault if the association query contains a value for the
710 AssociationClassName, Role, ResultClassName, or ResultRole method parameters that names a CIM
711 element that is not defined in the target CIM namespace or relevant CIM class.

712 The association query uses these parameters to filter the results and not to define the results.

713 Clients should use wsman:Filter when using IncludeResultProperty elements because these queries
714 contain projections and are not Boolean predicates.

715 **EXAMPLE:** The following request issues an association query in which the returned results include properties from
716 the associated instances as well as the EPRs of the associated instances. This example uses the
717 WS-Management Default Addressing Model but applies to any EPR model used by the service.

```

718 (1) <s:Envelope>
719 (2)   <s:Header>
720 (3)     <wsman:ResourceURI>
721 (4)       http://schemas.dmtf.org/wbem/wscim/1/*
722 (5)     </wsman:ResourceURI>
723 (6)   </s:Header>
724 (7)   <s:Body>
725 (8)     <wsen:Enumerate>
726 (9)       <wsman:EnumerationMode>EnumerateObjectAndEPR</wsman:EnumerationMode>
727 (10)      <wsman:Filter
728 (11)        Dialect="http://schemas.dmtf.org/wsman/cimbinding/associationFilter">
729 (12)          <wsmb:AssociatedInstances>
730 (13)            <wsmb:Object>
731 (14)              <wsa:Address> ... </wsa:Address>
732 (15)              <wsa:ReferenceParameters>
733 (16)                <wsman:ResourceURI>
734 (17)                  http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_PhysicalElement
735 (18)                </wsman:ResourceURI>
736 (19)                <wsman:SelectorSet>
737 (20)                  <wsman:Selector Name="Tag">81190b2</wsman:Selector>
738 (21)                  <wsman:Selector Name="CreationClassName">
739 (22)                    Vendor_PhysicalElement
740 (23)                  </wsman:Selector>
741 (24)                </wsman:SelectorSet>
742 (25)              </wsa:ReferenceParameters>
743 (26)            </wsmb:Object>
744 (27)            <wsmb:AssociationClassName>
745 (28)              CIM_SystemPackaging
746 (29)            </wsmb:AssociationClassName>
747 (30)            <wsmb:ResultClassName>CIM_System</wsmb:ResultClassName>
748 (31)            <wsmb:IncludeResultProperty>Name</wsmb:IncludeResultProperty>
749 (32)            <wsmb:IncludeResultProperty>
750 (33)              PrimaryOwnerName

```

```

751 (34)         </wsmb:IncludeResultProperty>
752 (35)         </wsmb:AssociatedInstances>
753 (36)         </wsman:Filter>
754 (37)         </wsen:Enumerate>
755 (38)         </s:Body>
756 (39)         </s:Envelope>

```

The results include the two requested properties as well as the EPR of the associated instances:

```

757 (40) <s:Body>
758 (41)   <wsen:PullResponse>
759 (42)     <wsen:EnumerationContext> ... </wsen:EnumerationContext>
760 (43)     <wsen:Items>
761 (44)       <wsman:Item>
762 (45)         <wsman:XmlFragment>
763 (46)           <Name>system1</Name>
764 (47)           <PrimaryOwnerName>Joe</PrimaryOwnerName>
765 (48)         </wsman:XmlFragment>
766 (49)         <wsa:EndpointReference>
767 (50)           <wsa:Address> ... </wsa:Address>
768 (51)           <wsa:ReferenceParameters>
769 (52)             <wsman:ResourceURI>
770 (53)               http://schemas.dmtf.org/cim/wscim/1/cim-schema/2/CIM_ComputerSystem
771 (54)             </wsman:ResourceURI>
772 (55)             ...
773 (56)           </wsa:ReferenceParameters>
774 (57)         </wsa:EndpointReference>
775 (58)       </wsman:Item>
776 (59)       <wsman:Item>
777 (60)         <wsman:XmlFragment>
778 (61)           <Name>system2</Name>
779 (62)           <PrimaryOwnerName>Mary</PrimaryOwnerName>
780 (63)         </wsman:XmlFragment>
781 (64)         <wsa:EndpointReference>
782 (65)           <wsa:Address> ... </wsa:Address>
783 (66)           <wsa:ReferenceParameters>
784 (67)             <wsman:ResourceURI>
785 (68)               http://schemas.vendor.com/.../Vendor_System
786 (69)             </wsman:ResourceURI>
787 (70)             ...
788 (71)           </wsa:ReferenceParameters>
789 (72)         </wsa:EndpointReference>
790 (73)       </wsman:Item>
791 (74)       ...etc.
792 (75)     </wsen:Items>
793 (76)   </wsen:PullResponse>
794 (77) </s:Body>

```

796 8.2.2 Association Instances

797 For queries that return instances of the association class used in a relationship, the Enumerate message
798 has the following form:

```
799 (1) <wsen:Enumerate>
800 (2)   <wsman:Filter
801 (3)     Dialect="http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter">
802 (4)     <wsmb:AssociationInstances>
803 (5)       <wsmb:Object> xs:any </wsmb:Object>
804 (6)       <wsmb:ResultClassName> xs:NCName </wsmb:ResultClassName> ?
805 (7)       <wsmb:Role> xs:NCName </wsmb:Role> ?
806 (8)       <wsmb:IncludeResultProperty> xs:NCName </wsmb:IncludeResultProperty> *
807 (9)     </wsmb:AssociationInstances>
808 (10)   </wsman:Filter>
809 (11) </wsen:Enumerate>
```

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

- 811 • wsen:Enumerate/wsman:Filter/wsmb:AssociationInstances

812 The results include association instances related to the source object.

813 **R8.2.2-1:** The results of the enumeration shall be instances of an association class subject to the
814 additional constraints listed in this clause.

- 815 • wsen:Enumerate/wsman:Filter/wsmb:AssociationInstances/wsmb:Object

816 Identifies the source object for the association query and is required.

817 **R8.2.2-2:** The results shall be instances of association classes for which one of the references is the
818 object identified by this endpoint reference.

819 **R8.2.2-3:** If the EPR to which the Enumerate message is sent and the EPR of the source object
820 represent two different CIM namespaces, the service may return a wsen:CannotProcessFilter fault.

821 **R8.2.2-4:** If the EPR of the source object does not reference exactly one valid CIM instance, the
822 service shall respond with a wsen:CannotProcessFilter fault. Services should include a textual
823 description of the problem.

- 824 • wsen:Enumerate/wsman:Filter/wsmb:AssociationInstances/wsmb:ResultClassName

825 Represents the name of a CIM association class. This element or parameter is optional.

826 **R8.2.2-5:** If the ResultClassName is present, the results shall contain only instances of the named
827 class or a derived class.

- 828 • wsen:Enumerate/wsman:Filter/wsmb:AssociationInstances/wsmb:Role

829 Represents the name of a reference property of a CIM association class. This element or parameter is
830 optional.

831 **R8.2.2-6:** If the Role element is present, the results shall include only instances of association classes
832 that refer to the source object through a property whose name matches the value of this parameter.

- 833 • wsen:Enumerate/wsman:Filter/wsmb:AssociationInstances/wsmb:IncludeResultProperty

834 Represents the name of one or more properties of a CIM class. This element or parameter is optional.

835 **R8.2.2-7:** If the query does not include an IncludeResultProperty element, the service shall return each
836 instance representation using the GED defined for the object's class within the wsen:Items element.

837 **R8.2.2-8:** If the query includes one or more IncludeResultProperty elements, the service shall return
 838 each instance representation using the wsman:XmlFragment element. Within the wsman:XmlFragment
 839 element, the service shall return property values using the property GEDs defined in the [WS-CIM](#)
 840 [Mapping Specification](#). If the query includes one or more IncludeResultProperty elements, the service
 841 shall not return any IncludeResultProperty elements not specified. The service shall ignore any
 842 IncludeResultProperty elements that describe properties not defined by the target class. If the service
 843 does not support fragment-level access, it shall return a wsman:UnsupportedFeature fault with the
 844 following detail code:

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

846 **R8.2.2-9:** A service may omit returned properties, even if explicitly requested, if and only if such
 847 properties have not been set (that is, the properties have a NULL value). The requestor is to interpret
 848 the absence of these properties as the properties having a value of NULL.

849 **R8.2.2-10:** A service shall not return a fault if the association query contains a value for the Role or
 850 ResultClassName method parameters that name a CIM element that is not defined in the target CIM
 851 namespace or relevant CIM class.

852 Clients should use wsman:Filter when using IncludeResultProperty elements as these queries contain
 853 projections and are not Boolean predicates.

854 9 Enumeration

855 The [WS-Enumeration](#) specification is used as a basis for iteration through the members of a collection.
 856 When enumerating instances of classes, the WS-Management Enumerate operation is used.

857 9.1 EnumerationMode

858 Supporting wsman:EnumerationMode enables clients to use enumeration as a method to discover
 859 instances. Clients can incorporate one of the EnumerationMode values to obtain the endpoint reference to
 860 such instances.

861 **R9.1-1:** To maximize interoperability, it is recommended that services that support enumeration also
 862 support wsman:EnumerationMode as defined in WS-Management.

863 **EXAMPLE 1:** The following example shows an unfiltered enumeration of a class. The class-specific ResourceURI is
 864 used when performing a simple unfiltered enumeration:

```
865 (1) ...
866 (2) <s:Header>
867 (3)   <wsa:Action>
868 (4)     http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate
869 (5)   </wsa:Action>
870 (6)
871 (7)   <wsman:ResourceURI>
872 (8)     http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ComputerSystem
873 (9)   </wsman:ResourceURI>
874 (10) </s:Header>
875 (11) <s:Body>
876 (12)   <wsen:Enumerate/>
877 (13) </s:Body>
```

878 Enumerating this ResourceURI returns all instances of the named class and any derived classes:

```
879 (1) <CIM_ComputerSystem> <Name>Red-202</Name> ... </CIM_ComputerSystem>
880 (2) <CIM_ComputerSystem> <Name>Blue-03</Name> ... </CIM_ComputerSystem>
881 (3) <CIM_ComputerSystem> <Name>Blue-04</Name> </CIM_ComputerSystem>
882 (4) <Vendor_ComputerSystem> <Name>Green-1</Name> ... </Vendor_ComputerSystem>
```

883 Each XML instance retrieved by the preceding enumeration contains all the properties of the specific
884 class. For example, the third XML instance is actually of type CIM_UnitaryComputerSystem and might
885 look as follows:

```
886 (1) <CIM_UnitaryComputerSystem
887 (2) xmlns= "http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_UnitaryComputerSystem">
888 (3)
889 (4) <Name> Blue-04 </Name>
890 (5) <PowerManagementSupported> true </PowerManagementSupported>
891 (6) <PrimaryOwnerName> Dave </PrimaryOwnerName>
892 (7) ...
893 (8)
894 (9) </CIM_UnitaryComputerSystem>
```

895 9.2 XmlFragment

896 XPath allows fragments of the instance to be returned.

897 **R9.2-1:** Some filter expressions allow fragments of the instance to be returned. When these ad-hoc
898 queries are performed, the results should be wrapped using wsman:XmlFragment as per R6.6-1 of the
899 *WS-Management Specification*.

900 EXAMPLE 1: The following filter expression finds the name of all CIM_ComputerSystems owned by Dave and
901 returns just the Name element of the instance provided that the owner is "Dave":

```
902 XPath: ../CIM_ComputerSystem[PrimaryOwnerName="Dave"]/Name
```

903 The filter expression results in a PullResponse of the following form:

```
904 (1) <wsen:PullResponse>
905 (2) <wsman:XmlFragment>
906 (3) <Name> Red-202 </Name>
907 (4) </wsman:XmlFragment>
908 (5) <wsman:XmlFragment>
909 (6) <Name> Blue-04 </Name>
910 (7) </wsman:XmlFragment>
911 (8) ...
912 (9) </wsen:PullResponse>
```

913 EXAMPLE 2: As a further refinement, just the value alone may be returned:

```
914 XPath: ../CIM_ComputerSystem[PrimaryOwnerName="Dave"]/Name/text()
```

915 This modification of the filter expression results in a PullResponse of the following form:

```
916 (1) <wsen:PullResponse>
917 (2) <wsman:XmlFragment> Red-202 </wsman:XmlFragment>
918 (3) <wsman:XmlFragment> Blue-04 </wsman:XmlFragment>
919 (4) ...
920 (5) </wsen:PullResponse>
```

921 9.3 Polymorphism

922 Many CIM implementations allow polymorphism.

923 A common way to extend CIM classes is to define derivatives of the CIM class. When a client requests
924 objects of the type for CIM_Process, it is possible to return instances that are actually of a derived type
925 such as Vendor_Process.

926 The result set may contain instances in accord with one of these three scenarios:

- 927 • Results should contain instances from the base class and all derived classes, and each instance
928 should be represented in its actual type including any derived properties.
- 929 • Results should contain instances from the base class and all derived classes, but the XML
930 document should be of the base class type and contain only elements corresponding to the
931 properties of the base class.
- 932 • Results should contain only instances of the base class and no instances of derived classes.

933 The default behavior is to return all instances in their native representation.

934 **R9.3-1:** A service supporting enumeration shall include instances from the requested class and derived
935 classes in the enumeration result unless otherwise directed by the client.

936 The client can request other behavior by adding the optional wsmb:PolymorphismMode element as a child
937 element of the wsen:Enumerate element in the Enumeration request, as follows:

```
938 (
939   ...
940   (
941     <s:Body>
942     (
943       <wsen:Enumerate>
944       (
945         ...
946         (
947           <wsmb:PolymorphismMode> ... </wsmb:PolymorphismMode> ?
948         )
949       </wsen:Enumerate>
950     )
951   </s:Body>
```

952 **R9.3-2:** A service may optionally support the wsmb:PolymorphismMode modifier element with a value
953 of ExcludeSubClassProperties. The ExcludeSubClassProperties PolymorphismMode shall return
954 instances of the requested class and derived classes represented using the base class's GED and
955 XSD type. Properties defined in the derived class are not returned.

956 **R9.3-3:** A service may optionally support the wsmb:PolymorphismMode modifier element with a value
957 of None. The None Polymorphism mode shall return instances of the requested class only.

958 **R9.3-4:** A service may optionally support the wsmb:PolymorphismMode modifier element with a value
959 of IncludeSubClassProperties. The IncludeSubClassProperties shall return instances of the requested
960 class and derived classes using the actual class's GED and XSD type. This is the same as not
961 specifying the polymorphism mode.

962 **R9.3-5:** If the service does not support the requested polymorphism mode, it should return a
963 wsmb:PolymorphismModeNotSupported fault.

964 **R9.3-6:** The service should return a wsmb:PolymorphismModeNotSupported fault for requests using
965 the "all classes" ResourceURI if the PolymorphismMode element is present and does not have a value
966 of IncludeSubClassProperties.

967 **R9.3-7:** If both wsman:EnumerationMode and wsmb:PolymorphismMode are supported and
 968 wsman:EnumerationMode is present in the request, the service shall always use the Resource URI of
 969 the actual class in the returned EPR regardless of the value of wsmb:PolymorphismMode. This allows
 970 the client to retrieve and update the actual instance.

971 **EXAMPLE 1:** The following example shows an unfiltered enumeration using just base class properties. Using the
 972 PolymorphismMode element along with the class-specific ResourceURI yields the same results as the
 973 example in 9.1, but the derived type is “cast away” or dropped.

```

974 (1) ...
975 (2) <s:Header>
976 (3)   <wsa:Action>
977 (4)     http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate
978 (5)   </wsa:Action>
979 (6)
980 (7)   <wsman:ResourceURI>
981 (8)     http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ComputerSystem
982 (9)   </wsman:ResourceURI>
983 (10) </s:Header>
984 (11) <s:Body>
985 (12)   <wsen:Enumerate>
986 (13)     <wsmb:PolymorphismMode> ExcludeSubClassProperties </wsmb:PolymorphismMode>
987 (14)   </wsen:Enumerate>
988 (15) </s:Body>

```

989 The same four instances are returned but “cast” as CIM_ComputerSystem:

```

990 (1) <CIM_ComputerSystem> <Name>Red-202</Name> ... </CIM_ComputerSystem>
991 (2) <CIM_ComputerSystem> <Name>Blue-03</Name> ... </CIM_ComputerSystem>
992 (3) <CIM_ComputerSystem> <Name>Blue-04</Name> ... </CIM_ComputerSystem>
993 (4) <CIM_ComputerSystem> <Name>Green-1</Name> ... </CIM_ComputerSystem>

```

994 Note that the third instance no longer contains the PowerManagementSupported property added by
 995 CIM_UnitaryComputerSystem:

```

996 (1) <CIM_ComputerSystem
997 (2)   xmlns="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ComputerSystem">
998 (3)
999 (4)   <Name> Blue-04 </Name>
1000 (5)   <PrimaryOwnerName> Dave </PrimaryOwnerName>
1001 (6)   ...
1002 (7)
1003 (8) </CIM_ComputerSystem>

```

1004 **R9.3-8:** If an Enumerate request specifies wsmb:PolymorphismMode=ExcludeSubClassProperties and
 1005 wsman:EnumerationMode=EnumerateObjectAndEPR or EnumerateEPR, then the service shall return
 1006 EPRs that reference instances of the most-derived classes of the requested class in the ResourceURI.

1007 The body of the request message appears as follows:

```

1008 (1) <wsen:Enumerate>
1009 (2)   <wsman:EnumerationMode> EnumerateObjectAndEPR </wsman:EnumerationMode>
1010 (3)   <wsmb:PolymorphismMode> ExcludeSubClassProperties </wsmb:PolymorphismMode>
1011 (4) </wsen:Enumerate>

```

1012 The corresponding response message contains the following fragment. Note that the EPR for Blue-04
 1013 can be used to access the property PrimaryOwnerName that is not present in the value returned.

```

1014 (1) <wsen:Items>
1015 (2)   <wsman:Item>
1016 (3)     <CIM_ComputerSystem <Name>Red-202</Name> ... </CIM_ComputerSystem>
1017 (4)     <wsa:EndpointReference>
1018 (5)       <wsa:Address> ... </wsa:Address>
1019 (6)       <wsa:ReferenceParameters>
1020 (7)         <wsman:ResourceURI>
1021 (8)           http://schemas.dmtf.org/.../CIM_ComputerSystem
1022 (9)         </wsman:ResourceURI>
1023 (10)        <wsman:SelectorSet> ... </wsman:SelectorSet>
1024 (11)        </wsa:ReferenceParameters>
1025 (12)       </wsa:EndpointReference>
1026 (13)     </wsman:Item>
1027 (14)   <wsman:Item>
1028 (15)     <CIM_ComputerSystem <Name>Blue-04</Name> ... </CIM_ComputerSystem>
1029 (16)     <wsa:EndpointReference>
1030 (17)       <wsa:Address> ... </wsa:Address>
1031 (18)       <wsa:ReferenceParameters>
1032 (19)         <wsman:ResourceURI>
1033 (20)           http://schemas.dmtf.org/.../CIM_UnitaryComputerSystem
1034 (21)         </wsman:ResourceURI>
1035 (22)        <wsman:SelectorSet> ... </wsman:SelectorSet>
1036 (23)        </wsa:ReferenceParameters>
1037 (24)       </wsa:EndpointReference>
1038 (25)     </wsman:Item>
1039 (26)     ...
1040 (27) </wsen:Items>

```

1041 **9.4 XPath Enumeration Using the Class-Specific ResourceURI**

1042 The ResourceURI contains the class name, as for unfiltered enumeration:

```

1043 (1) <wsman:ResourceURI>
1044 (2)   http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ComputerSystem
1045 (3) </wsman:ResourceURI>

```

1046 The XPath is anchored at an abstract array of CIM_ComputerSystem XML nodes, which represent all
 1047 available instances:

```

1048 (1) <CIM_ComputerSystem> ... </CIM_ComputerSystem>
1049 (2) <CIM_ComputerSystem> ... </CIM_ComputerSystem>
1050 (3) <CIM_ComputerSystem> ... </CIM_ComputerSystem>
1051 (4) <CIM_ComputerSystem> ... </CIM_ComputerSystem>

```

1052 The XPath filter expression is evaluated against each possible instance of the specified class, and the
 1053 instance is either selected as part of the result set or is discarded.
 1054 PolymorphismMode=ExcludeSubClassProperties is used to ensure that all instances have the same type.

1055 The following XPath expressions all select every instance of CIM_ComputerSystem and are identical:

```

1056 (1) XPath: .
1057 (2) XPath: ../CIM_ComputerSystem

```

1058 To filter, the [] filter expressions from XPath may be used. The following selects only instances that have a
 1059 PrimaryOwnerName property set to "Dave":

1060 `XPath: ../CIM_ComputerSystem[PrimaryOwnerName="Dave"]`

1061 If PolymorphismMode=IncludeSubClassProperties were used, the following two XPath filters would have
 1062 different results:

1063 (1) `XPath: .[Owner="Dave"]`

1064 (2) `XPath: ../CIM_ComputerSystem[Owner="Dave"]`

1065 The first XPath would match all instances regardless of type, while the second XPath would select only
 1066 those instances whose actual type was CIM_ComputerSystem.

1067 9.5 XPath Enumerate Using the "All Classes" ResourceURI

1068 As an alternative to a class-specific ResourceURI, the URI meaning "all classes" may be specified:

1069 `http://schemas.dmtf.org/wbem/wscim/1/*`

1070 This URI is a resource that refers to all instances of all classes. In this case, the abstract array of instances
 1071 is mixed and includes elements of classes other than CIM_ComputerSystem.

1072 (1) `<CIM_ComputerSystem> ... </CIM_ComputerSystem>`

1073 (2) `<CIM_ComputerSystem> ... </CIM_ComputerSystem>`

1074 (3) `<CIM_SoftwareElement> ... </CIM_SoftwareElement>`

1075 (4) `<CIM_SoftwareElement> ... </CIM_SoftwareElement>`

1076 (5) `<CIM_LogicalDisk> ... </CIM_LogicalDisk>`

1077 (6) `<CIM_LogicalDisk> ... </CIM_LogicalDisk>`

1078 (7) `<CIM_LogicalDisk> ... </CIM_LogicalDisk>`

1079 (8) `...etc.`

1080 In the following example, the first query contains no class-specific information. Therefore, the query
 1081 specifies "all instances of all classes". The second query refers to a specific class:

1082 (1) `XPath: .`

1083 (2) `XPath: ../CIM_ComputerSystem`

1084 Services do not typically support the first query if the "all classes" ResourceURI is used, but they may do
 1085 so.

1086 NOTE: The XPath queries are identical to those provided in 9.4. The ResourceURI simply changes the implied pool
 1087 of instances over which the query is executed.

1088 10 Subscriptions

1089 The WS-Management Subscribe operation (from WS-Eventing) is used to subscribe to CIM indications.
 1090 WS-Eventing uses the term "event" for the SOAP message sent to the receiver, while CIM uses the term
 1091 "indication" for the observation of an event.

1092 The CIM Schema defines a set of special classes to support the delivery of indications to interested
 1093 receivers. In the CIM Schema, indications are represented by the CIM_Indication class or a subclass of
 1094 CIM_Indication. Subscriptions can express interest in a set of CIM_Indications by providing a query
 1095 expression or by referring to an already existing query. This clause outlines the relationship between the
 1096 WS-Eventing messages and these CIM classes.

1097 A typical scenario for use of CIM indications would be a management client interested in receiving "sensor
1098 state change" indications from a device that it is managing. To receive these indications, the client would
1099 take the following steps:

- 1100 1) Construct or identify the indication filter.
- 1101 2) Create the WS-Eventing Subscribe request.
- 1102 3) Receive indications.

1103 A management service might need the ability to report on all subscriptions on a server.

1104 In the CIM Schema, subscriptions are represented by a trio of classes:

- 1105 • CIM_IndicationFilter (or CIM_FilterCollection) captures the query or filter identifying the subset of
1106 indications of interest.
- 1107 • CIM_ListenerDestination captures information about where or how the indications are to be
1108 delivered.
- 1109 • CIM_IndicationSubscription (or CIM_FilterCollectionSubscription) associates an instance of
1110 CIM_IndicationFilter (or CIM_FilterCollection) with CIM_ListenerDestination.

1111 These classes are used in different parts of the subscription life cycle, as indicated in the remainder of this
1112 clause.

1113 **R10-1:** A service that supports subscriptions shall do so using the WS-Eventing operations as defined
1114 in WS-Management. It is recommended that a service internally create the requisite CIM indication-
1115 related instances when the service accepts a subscription using the Subscribe message from a Web
1116 services client.

1117 **R10-2:** A service may deliver indications based on the creation of instances of the CIM indication-
1118 related classes in addition to supporting WS-Eventing.

1119 **R10-3:** A service that does not support the WS-Management Default Addressing Model is not required
1120 to conform to the rules for the ResourceURI described in the text and examples in the following
1121 subclauses (clause 10 and its subclauses). All examples about WS-Eventing filter dialects apply to
1122 services independent of their addressing model.

1123 10.1 Indication Filters

1124 When subscribing to indications, the same XPath and CQL filter usage is observed as for enumerations.
1125 However, association queries are not applicable to subscriptions.

1126 When CQL is used, the subscription filter includes the name of the class being selected for the
1127 subscription:

```
1128 select * from CIM_AlertIndication where MessageID="394"
```

1129 CQL statements with projections can also be used, in which case the selected properties of the indications
1130 are wrapped using wsman:XmlFragment as described in 8.1.

1131 The same filter can be expressed in XPath:

```
1132 ../../CIM_AlertIndication[MessageID="394"]
```

1133 XPath filters can also be written without identifying the class. The same filter could be expressed using the
1134 following XPath filter if it were applied to instances of CIM_AlertIndication:

```
1135 ../../[MessageID="394"]
```

1136 These filter expressions can be formulated by the client, or they might already exist on the server (as an
1137 instance of CIM_IndicationFilter).

1138 10.2 Subscribe Request

1139 The client constructs the subscribe request to express interest in a subset of the indications on the service.
1140 The client can filter the indications by specifying a filter directly in the subscribe request or by referring to an
1141 existing filter stored on the service.

1142 10.2.1 Subscribing Using a Filter

1143 When subscribing using a filter expression, the client can target the subscribe request to either the CIM
1144 Server or a specific indication class.

1145 10.2.1.1 Subscribing to the CIM Server

1146 When subscribing to the CIM Server, a filter dialect such as CQL can be used. In this case, the query alone
1147 contains the necessary information as to which class is being filtered and the “all classes” ResourceURI
1148 can be used for addressing.

1149 **R10.2.1.1-1:** If a service supports client-supplied CQL expressions and the WS-Management Default
1150 Addressing Model, it should accept wse:Subscribe messages addressed to the “all-classes”
1151 ResourceURI.

1152 **EXAMPLE:** The following example shows a Subscribe message to set up a subscription for changes in sensor state.
1153 It is addressed to the “all classes” ResourceURI and uses a CQL filter to detect instance indications in
1154 which the CurrentState property has changed:

```
1155 (1) <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1156 (2)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1157 (3)   xmlns:wsmn="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
1158 (4)   xmlns:wse="http://schemas.xmlsoap.org/ws/2004/09/eventing">
1159 (5) <s:Header>
1160 (6)   <wsa:Action>
1161 (7)     http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
1162 (8)   </wsa:Action>
1163 (9)   <wsa:To> http://127.0.0.1:9999/wsman </wsa:To>
1164 (10)  <wsa:MessageID> . . . </wsa:MessageID>
1165 (11)  <wsa:ReplyTo>
1166 (12)    http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
1167 (13)  </wsa:ReplyTo>
1168 (14)  <wsmn:ResourceURI>
1169 (15)    http://schemas.dmtf.org/wbem/wscim/1/*
1170 (16)  </wsmn:ResourceURI>
1171 (17) </s:Header>
1172 (18) <s:Body>
1173 (19)   <wse:Subscribe>
1174 (20)     <wse:Delivery
1175 (21)       Mode="http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck">
1176 (22)     <wse:NotifyTo>
1177 (23)       <wsa:Address> . . . </wsa:Address>
1178 (24)       . . .
1179 (25)     </wse:NotifyTo>
1180 (26)   </wse:Delivery>
1181 (27)   <wsmn:Filter dialect="http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf">
```



```

1182 (28)      <!-- whenever the state of any sensor changes -->
1183 (29)      SELECT *
1184 (30)      FROM CIM_InstIndication
1185 (31)      WHERE SourceInstance ISA CIM_Sensor
1186 (32)      AND PreviousInstance ISA CIM_Sensor
1187 (33)      AND PreviousInstance.CIM_Sensor::CurrentState &lt;&gt;
1188 (34)      SourceInstance.CIM_Sensor::CurrentState
1189 (35)      </wsman:Filter>
1190 (36)      </wse:Subscribe>
1191 (37) </s:Body>
1192 (38) </s:Envelope>

```

1193 When subscribing to the CIM Server, instances of all classes are implicitly addressed; therefore, separate
 1194 polymorphism modes are not relevant.

1195 **R10.2.1.1-2:** A service supporting wse:Subscribe messages addressed to the “all classes”
 1196 ResourceURI shall return a wsmb:PolymorphismModeNotSupported fault if the
 1197 wsmb:PolymorphismMode modifier is present and does not equal IncludeSubClassProperties.

1198 10.2.1.2 Subscribing to an Indication Class

1199 A subset of all indications can also be expressed by subscribing to an indication class. In this case, the
 1200 EPR contains the necessary information as to which class is being filtered. An additional filter might or
 1201 might not be present, but it would apply only to the instances of class indicated by the EPR.

1202 **R10.2.1.2-1:** If a service supports client filtering over a particular class of indications and the
 1203 WS-Management Default Addressing Model, it should accept wse:Subscribe messages addressed to
 1204 the class-specific ResourceURI for CIM_Indication or a subclass of CIM_Indication.

1205 **EXAMPLE:** The following example shows a Subscribe message to set up a subscription for changes in temperature
 1206 sensors. It is addressed to the resource URI for the CIM_AlertIndication class and uses XPath to select
 1207 instances of the class in which one of the desired messages is present:

```

1208 (1) <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1209 (2)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1210 (3)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
1211 (4)   xmlns:wse="http://schemas.xmlsoap.org/ws/2004/09/eventing" >
1212 (5) <s:Header>
1213 (6)   <wsa:Action>
1214 (7)     http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
1215 (8)   </wsa:Action>
1216 (9)   <wsa:To> http://127.0.0.1:9999/wsman </wsa:To>
1217 (10)  <wsa:MessageID> . . . </wsa:MessageID>
1218 (11)  <wsa:ReplyTo>
1219 (12)    http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
1220 (13)  </wsa:ReplyTo>
1221 (14)  <wsman:ResourceURI>
1222 (15)    http://schemas.dmtf.org/wbem/wscim/1/CIM_AlertIndication
1223 (16)  </wsman:ResourceURI>
1224 (17) </s:Header>
1225 (18) <s:Body>
1226 (19)   <wse:Subscribe>
1227 (20)     <wse:Delivery
1228 (21)       Mode="http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck">
1229 (22)     <wse:NotifyTo>

```

```

1230 (23)         <wsa:Address> . . . </wsa:Address>
1231 (24)         . . .
1232 (25)         </wse:NotifyTo>
1233 (26)         </wse:Delivery>
1234 (27)         <wsman:Filter
1235 (28)             xmlns:c="http://schemas.dmtf.org/wbem/wscim/1/CIM_AlertIndication">
1236 (29)             .[c:OwningEntity="DMTF" and (c:MessageID="394" or c:MessageID="396"
1237 (30)             or c:MessageID="398" or c:MessageID="400" or c:MessageID="413")]
1238 (31)         </wsman:Filter>
1239 (32)     </wse:Subscribe>
1240 (33) </s:Body>
1241 (34) </s:Envelope>

```

1242 Additional filtering, such as XPath filters, on the instances of CIM_AlertIndication that are identified by the
 1243 EPR can be allowed. However, this practice is discouraged because using CQL expressions in this context
 1244 creates the possibility for contradictions between the class identified by the EPR and the class identified in
 1245 the CQL expression.

1246 **R10.2.1.2-2:** A service that supports a class-specific ResourceURI as a target of the wse:Subscribe
 1247 message should return the wse:InvalidMessage fault if such messages specify a filter that includes
 1248 class information as part of the filter expression.

1249 When the wse:Subscribe message is addressed to an indication class, the wsmb:PolymorphismMode
 1250 element described in 9.3 can be used to control how polymorphism is handled for indications on event
 1251 delivery. The wsmb:PolymorphismMode element becomes a child element of the Subscribe element.

1252 **R10.2.1.2-3:** A service supporting wse:Subscribe messages addressed to a CIM indication class
 1253 through a class-specific ResourceURI shall provide indication instances from the requested class and
 1254 its subclasses in event delivery unless otherwise directed by the client.

1255 **R10.2.1.2-4:** A service supporting wse:Subscribe messages addressed to a CIM indication class
 1256 through a class-specific ResourceURI may support the use of the wsmb:PolymorphismMode modifier
 1257 as a child of the wse:Subscribe element, with the resulting event instances typed according to rules
 1258 **R9.3-2**, **R9.3-3**, and **R9.3-4**.

1259 10.2.2 Subscribing to an Existing Filter

1260 The service may have existing filters because of profile provisions implemented or filters previously created
 1261 by a client. The client needs a way to express interest in one of these filters. These filters are represented
 1262 by instances of either the CIM_IndicationFilter or CIM_FilterCollection classes; hereafter these instances
 1263 are referred to as existing filters.

1264 **R10.2.2-1:** If a service supports filtering using an existing filter expression and the WS-Management
 1265 Default Addressing Model, it should accept wse:Subscribe messages addressed to the class-specific
 1266 ResourceURI for an instance of the existing filter class.

1267 **EXAMPLE:** The following example shows a Subscribe message to set up a subscription to an existing filter named by
 1268 "example.org::temperatureSensors::stateChanges":

```

1269 (1) <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1270 (2)     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1271 (3)     xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
1272 (4)     xmlns:wse="http://schemas.xmlsoap.org/ws/2004/09/eventing" >
1273 (5) <s:Header>
1274 (6)     <wsa:Action>
1275 (7)         http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
1276 (8)     </wsa:Action>

```

```

1277 (9) <wsa:To> http://127.0.0.1:9999/wsman </wsa:To>
1278 (10) <wsa:MessageID> . . . </wsa:MessageID>
1279 (11) <wsa:ReplyTo>
1280 (12) http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
1281 (13) </wsa:ReplyTo>
1282 (14) <wsman:ResourceURI>
1283 (15) http://schemas.dmtf.org/wbem/wscim/1/CIM_IndicationFilter
1284 (16) </wsman:ResourceURI>
1285 (17) <wsman:SelectorSet>
1286 (18) <wsman:Selector name="Name">
1287 (19) example.org::temperatureSensors::stateChanges
1288 (20) </wsman:Selector>
1289 (21) <wsman:Selector name="SystemCreationClassName">
1290 (22) CIM_ComputerSystem
1291 (23) </wsman:Selector>
1292 (24) <wsman:Selector name="__cimnamespace">interop</wsman:Selector>
1293 (25) </wsman:SelectorSet>
1294 (26) </s:Header>
1295 (27) <s:Body>
1296 (28) <wse:Subscribe>
1297 (29) <wse:Delivery
1298 (30) Mode="http://schemas.dmtf.org/wbem/wsman/1/wsman/PushWithAck">
1299 (31) <wse:NotifyTo>
1300 (32) <wsa:Address> . . . </wsa:Address>
1301 (33) . . .
1302 (34) </wse:NotifyTo>
1303 (35) </wse:Delivery>
1304 (36) <!-- wse:Filter and wsman:Filter not permitted in this case. -->
1305 (37) </wse:Subscribe>
1306 (38) </s:Body>
1307 (39) </s:Envelope>

```

1308 **R10.2.2-2:** If a service supports filtering using an existing filter expression (as indicated by the EPR),
1309 the service message shall return the wsman:InvalidParameter fault if the wse:Subscribe request
1310 includes a filter expression (such as in the wse:Filter or wsman:Filter elements).

1311 **R10.2.2-3:** A service supporting Subscribe to an existing filter using the WS-Management Default
1312 Addressing Model should support access using a class-specific ResourceURI corresponding to a filter
1313 with selector values that identify the instance of the actual class of the desired filter. The referenced
1314 base class shall be one for which CIM keys have been defined; otherwise, the service should respond
1315 with a wsman:InvalidSelectors fault with the following detail code:

1316 <http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/UnexpectedSelectors>

1317 When subscribing to an existing filter, the classes of interest are indicated by the filter expression and
1318 separate polymorphism modes are not relevant.

1319 **R10.2.2-4:** A service supporting wse:Subscribe messages addressed to an instance of
1320 CIM_IndicationFilter or CIM_FilterCollection through a class-specific ResourceURI shall return a
1321 wsmb:PolymorphismModeNotSupported fault if the wsmb:PolymorphismMode modifier is present and
1322 does not equal IncludeSubClassProperties.

1323 Subscribing to an instance of CIM_IndicationFilter (or CIM_FilterCollection) works regardless of whether or
 1324 not the service created the filter or if a client constructed the instance prior to sending the Subscribe
 1325 message. The client can construct instances of these filter classes using mechanisms such as WS-Transfer
 1326 Create. In this case, the service is accepting a client-defined filter expression, so the service must also
 1327 accept the same filter expression in a Subscribe message.

1328 **R10.2.2-5:** If a service supports creating an instance of CIM_IndicationFilter (using WS-Transfer
 1329 Create or another mechanism), the service shall also support a wse:Subscribe message in which the
 1330 filter expression is specified in the wsman:Filter element in body of the Subscribe message.

1331 10.3 Subscription Response

1332 A successful SubscribeResponse message includes a SubscriptionManager element containing an EPR to
 1333 be used to Unsubscribe from or Renew this subscription.

1334 **R10.3-1:** The SubscriptionManager EPR in a successful SubscribeResponse shall be unique to the
 1335 subscription created by the Subscribe request.

1336 That is, the SubscriptionManager EPR returned by the service shall contain some elements that correlate
 1337 one-to-one with the single subscription that was just created.

1338 **R10.3-2:** A service shall accept an Unsubscribe or Renew request whose EPR matches a
 1339 SubscriptionManager EPR that was previously returned to a client, provided that the subscription is still
 1340 active.

1341 That is, if a service accepts a subscription and returns a SubscriptionManager EPR to a client, the service
 1342 shall accept that EPR as the target of an Unsubscribe or Renew message.

1343 Because both the client and the service depend on this EPR, the SubscriptionManager EPR shall be valid
 1344 for the duration of the subscription.

1345 10.4 Event Delivery

1346 When instances of CIM_Indication or a subclass are indicated by the eventing infrastructure, they are
 1347 delivered as event SOAP messages according to the delivery mode in the wse:Subscribe request. The
 1348 following rules describe the XML representation of the indication:

1349 **R10.4-1:** When delivering the event XML for an indication, the wsa:Action URI of the event should be
 1350 set to the same value as the XML namespace for the actual class of the indication instance.

1351 **R10.4-2:** When delivering the event XML for an indication, the event body shall be the XML
 1352 representation of the indication instance as per the [WS-CIM Mapping Specification](#), subject to any
 1353 additional client requests such as projection or polymorphism.

1354 EXAMPLE: The following example shows an instance of CIM_InstModification delivered as a single event using the
 1355 Push delivery mode:

```

1356 (1) <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
1357 (2)   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1358 (3)   xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
1359 (4)   xmlns:class=
1360 (5)     "http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_InstModification"
1361 (6)   xmlns:common="http://schemas.dmtf.org/wbem/wscim/1/common"
1362 (7)   xmlns:wse="http://schemas.xmlsoap.org/ws/2004/09/eventing">
1363 (8) <s:Header>
1364 (9)   <wsa:Action>
1365 (10)     http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_InstModification
1366 (11)   </wsa:Action>

```

```

1367 (12) <wsa:To> . . . </wsa:To>
1368 (13) <wsa:MessageID> . . . </wsa:MessageID>
1369 (14) </s:Header>
1370 (15) <s:Body>
1371 (16) <class:CIM_InstModification>
1372 (17) <class:IndicationIdentifier>
1373 (18) CIM:12345678-abcd-0000-fedc-0123456789ab
1374 (19) </class:IndicationIdentifier>
1375 (20) <class:IndicationTime>
1376 (21) <common:dateTime>2007-04-01T11:22:33.123Z</common:dateTime>
1377 (22) </class:IndicationTime>
1378 (23) <class:PerceivedSeverity>5</class:PerceivedSeverity>
1379 (24) <class:PreviousInstance> . . . </class:PreviousInstance>
1380 (25) <class:SourceInstance> . . . </class:SourceInstance>
1381 (26) <class:SourceInstanceHost>10.57.217.39</class:SourceInstanceHost>
1382 (27) <class:SourceInstanceModelPath> . . . </class:SourceInstanceModelPath>
1383 (28) </class:CIM_InstModification>
1384 (29) </s:Body>
1385 (30) </s:Envelope>

```

1386 10.5 Subscription Reporting

1387 Subscription Reporting is the ability of an implementation to report on the existing filters, collections, and
 1388 subscriptions. Subscriptions can be created and deleted through the Subscribe and Unsubscribe
 1389 operations. Filters and subscriptions may also be created, modified, and deleted directly using other
 1390 protocol operations described in this specification. An implementation should instantiate instances that
 1391 reflect the results of the operations described in this specification.

1392 **R10.5-1:** It is recommended that a service create in its CIM service the requisite CIM indication-related
 1393 instances when the service accepts a subscription using the Subscribe message from a Web services
 1394 client. The CIM namespace in which these instances are created is beyond the scope of this
 1395 specification.

1396 The rules in the following clauses describe requirements for the content of the CIM indication-related
 1397 classes if such reporting is supported as recommended in the preceding rule.

1398 Every active subscription contains three components:

- 1399 • An instance of CIM_IndicationFilter or CIM_FilterCollection that describes the indications to be
 1400 delivered;
- 1401 • An instance of CIM_ListenerDestinationWSManagement that describes the client-specified
 1402 endpoint for delivery of indications; and
- 1403 • An instance of CIM_IndicationSubscription or CIM_FilterCollectionSubscription that links the filter
 1404 and the destination, and describes additional characteristics of the subscription.

1405 10.5.1 CIM_IndicationFilter

1406 The CIM_IndicationFilter class captures the filter used in the subscription.

1407 **R10.5.1-1:** If a subscribe request contains a filter expression, a service shall create an instance of
 1408 CIM_IndicationFilter and set the properties as indicated in Table 2.

1409

Table 2 – CIM_IndicationFilter Properties

Property Name	Value
Query	Filter expression from the Subscribe request, including XML if appropriate for the indicated QueryLanguage
QueryLanguage	Dialect URI from the Subscribe request For example, if a CQL expression were used in the Subscribe request the URI would be: http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf

1410 When subscribing to an existing filter expression, the instance of CIM_IndicationFilter already exists so a
1411 new instance is not created.

1412 10.5.2 CIM_ListenerDestinationWSManagement

1413 The CIM_ListenerDestinationWSManagement class captures the endpoint for event delivery.

1414 **R10.5.2-1:** A service shall ensure that, for each subscribed endpoint, an instance of
1415 CIM_ListenerDestinationWSManagement exists and contains the properties as indicated in Table 3.

1416

Table 3 – CIM_ListenerDestinationWSManagement Required Properties

Property Name	Value
Protocol	4 ("WS-Management")
Destination	The URL in the wsa:Address element of wse:NotifyTo If the delivery mode does not have a destination EPR (such as the Pull delivery mode), the WS-Addressing anonymous URI should be used as a place holder. Using the anonymous URI indicates that the event sink will contact the event source; the anonymous URI is not to be confused with the ReplyTo EPR in that request.

1417 A WS-Management subscription contains a number of terms that extend the concept of a CIM subscription.
1418 Additional properties in CIM_ListenerDestinationWSManagement capture these extensions. In most cases,
1419 the values of the new properties come from elements in the Subscribe request. In a few cases, the values
1420 are dictated by the WS-Management protocol.

1421 These properties are likely to be managed by users and client applications, and they might be of interest to
1422 users enumerating existing subscriptions. Some small footprint implementations of WS-Management
1423 services might not wish to expose all these properties.

1424 **R10.5.2-2:** If the subscribe request specifies any of the following options, the corresponding
1425 properties of the CIM_ListenerDestinationWSManagement instance should be set according to the
1426 values shown in Table 4. These guidelines might be updated by newer versions of this class; the actual
1427 MOF definition takes precedence over the information in Table 4.

1428

Table 4 – CIM_ListenerDestinationWSManagement Optional Properties

Property Name	Value
DestinationEndTo	Similar to Destination, but applies to the EndTo EPR, if present
Locale	RFC 3066 language code from the Subscribe request, if present
ContentEncoding	The value of the ContentEncoding element from the Subscribe request, if present
DeliveryMode	A ValueMap value that captures the Delivery/@Mode URI from the Subscribe request

Property Name	Value
Heartbeat	Interval in seconds at which point a heartbeat event will be sent if no other events have been sent
SendBookmarks	True if the SendBookmarks element was present in the Subscribe request
MaxTime	The time in seconds to build a batch when using a batching delivery mode
DeliveryAuth	The security profile URI being used by the event source when delivering events through a Push delivery mode
PolymorphismMode	A ValueMap value that captures the polymorphism choice if present in the Subscribe request

1429 In general, instances of ListenerDestinationWSManagement are not reusable because of the terms of the
 1430 subscription and the rules regarding their deletion when a subscription ends. Whether instances are shared
 1431 is beyond the scope of this specification.

1432 **10.5.3 CIM_IndicationSubscription and CIM_FilterCollectionSubscription**

1433 The CIM_IndicationSubscription and CIM_FilterCollectionSubscription classes capture associations
 1434 between the indication filter or filter collection and the endpoint for event delivery. An instance of one of
 1435 these classes represents the subscription created by the Subscribe request.

1436 **R10.5.3-1:** If a Subscribe request is addressed to an instance of CIM_IndicationFilter, or results in
 1437 the creation of an instance of CIM_IndicationFilter, then a service shall create an instance of
 1438 CIM_IndicationSubscription and set the properties as indicated in Table 5 as part of a successful
 1439 Subscribe operation.

1440 **Table 5 – Required Properties for CIM_IndicationSubscription and CIM_FilterCollectionSubscription**

Property Name	Value
SubscriptionDuration	The time at which the subscription expires as indicated in the Subscribe response
OnFatalErrorPolicy = "Remove"	Not applicable
RepeatNotificationPolicy = "None"	Not applicable
SubscriptionInfo	Unique value identifying the subscription

1441 **R10.5.3-2:** If a subscription request is addressed to an instance of CIM_FilterCollection, then a
 1442 service shall instead create an instance of CIM_FilterCollectionSubscription with properties as
 1443 indicated in Table 5.

1444 **R10.5.3-3:** If a service that supports Renew created an instance of CIM_IndicationSubscription (or
 1445 CIM_FilterCollectionSubscription) when processing the Subscribe message, it shall update the
 1446 SubscriptionDuration to reflect the new expiration time when processing the Renew message.

1447 WS-Eventing uses the subscription manager EPR in the SubscribeReponse message to identify the
 1448 subscription. It defines the wse:Identifier element for use as a reference parameter in this EPR, but it is not
 1449 required. For convenience, it is recommended that this element be used and match the SubscriptionInfo
 1450 property.

1451 **R10.5.3-4:** A service should populate the SubscriptionInfo field with a URI to identify the subscription
 1452 and use the same value as the value of the wse:Identifier reference parameter in the
 1453 SubscriptionManager EPR.

1454 Services can use the same URI format as outlined in 2.7 of the [WS-Management Specification](#) for
1455 wsa:MessageID.

1456 **10.5.4 Proxy Considerations**

1457 In some cases, the WS-Management service might be a proxy or adapter to an existing system. Such
1458 implementations have the following two pieces of information to track:

- 1459 • the information about the subscription between the client and the WS-Management service
- 1460 • the information about the subscription between the WS-Management service and the CIM Server

1461 The rules in this specification describe how to represent the information about the subscription between the
1462 client and the WS-Management service. The representation of the information between the
1463 WS-Management service and the CIM Server is beyond the scope of this specification.

1464 Implementations can choose to represent this “local” subscription using similar techniques, but the
1465 information would differ in properties such as the CIM_ListenerDestination.Destination that would be the
1466 address of the WS-Management service for the local subscription. Implementations can choose to create
1467 parallel subscriptions for each or do analysis to avoid sending the same indication multiple times on the
1468 local channel.

1469 **10.6 Unsubscribe and Renew Requests**

1470 A client may extend the duration of a subscription using a wse:Renew request, if the service supports such
1471 requests.

1472 **R10.6-1:** If a service supports eventing but does not support renewing subscriptions, the service may
1473 fault a wse:Renew request with the fault code wse:UnableToRenew. If a service supports eventing, the
1474 service shall not fault a wse:Renew request with fault code wsa:ActionNotSupported

1475 Unsubscribe and Renew requests may be addressed to a service using the SubscriptionManager EPR that
1476 was returned in the SubscribeResponse message.

1477 In lieu of using the SubscriptionManager EPR from the SubscribeResponse message, a client may
1478 construct a new SubscriptionManager EPR of a particular form that is acceptable to the service. If the
1479 ReferenceParameters of the EPR uniquely specify an existing instance of IndicationSubscription or
1480 FilterCollectionSubscription, a service is required to accept the Unsubscribe or Renew request at the
1481 normal protocol endpoint address, that is, the protocol endpoint where that subscription can be seen with
1482 Enumerate or Get. The To address of the SubscriptionManager EPR is not necessarily valid over long
1483 periods of time; the address may change because of dynamic addressing assigned to the protocol endpoint
1484 or subscription manager service.

1485 **R10.6-2:** A service shall accept an Unsubscribe request or Renew request whose EPR specifies a valid
1486 instance of IndicationSubscription or FilterCollectionSubscription. A service shall accept a request of
1487 this form at the To address of the protocol endpoint at which the subscription can be accessed with
1488 Enumerate or Get operations. A service may also accept a request of this form at the To address of the
1489 SubscriptionManager EPR.

1490 If the EPR does not specify a valid and unique IndicationSubscription or FilterCollectionSubscription, then
1491 the service shall fault the request. For instance, if a subscription has been terminated for any reason, then
1492 a SubscriptionManager EPR or a constructed EPR specifying that subscription will not be valid.

1493 **R10.6-3:** A service shall delete at most one subscription as a result of an Unsubscribe request.

1494 The Unsubscribe request shall be sufficiently specific that it removes one subscription, or none in the case
1495 of a fault for any reason.

1496 When a subscription is terminated, a service is required to clean up data structures that were created to
1497 represent the subscription.

1498 When a subscriber is no longer interested in receiving indications from a subscription, it can cancel the
1499 subscription using a wse:Unsubscribe request.

1500 **R10.6-4:** If a service created CIM indication-related instances as described in 10.5, then the service
1501 shall delete those instances when the subscription is canceled for any reason.

1502 In all cases, the instance of CIM_IndicationSubscription (or CIM_FilterCollectionSubscription) is deleted
1503 because this instance represents the actual subscription.

1504 Instances of the other members of the association might be reused between subscriptions. For example, if
1505 a subscription were addressed to an existing filter (an instance of CIM_IndicationFilter), then that instance
1506 need not be deleted when the subscription is deleted. The exact ownership of these instances and a
1507 method to determine when to delete them is beyond the scope of this specification.

1508 11 Extrinsic Methods

1509 Invoking an extrinsic method uses the action URIs and messages defined by the [WS-CIM Mapping](#)
1510 [Specification](#) (clause 8.3, "CIM Methods to WSDL Mappings"). The request and response message
1511 schemas for an extrinsic method are defined in the WS-CIM schema for the CIM class that defines the
1512 method (and the request and response message schemas use the XML namespace for that class). The
1513 wsa:Action URIs are derived from the XML namespace of the class and the method name as per the
1514 [WS-CIM Mapping Specification](#). The endpoint reference is transformed into SOAP headers as defined by
1515 WS-Addressing in the same way as other WS-Management operations.

1516 When using the WS-Management Default Addressing Model, the rules for ResourceURI and selector
1517 usage are the same as those described in clause 7 of this specification.

1518 12 Exceptions

1519 For some CIM server implementations, invoking either an intrinsic or extrinsic method can result in the
1520 production of one or more exceptions before the corresponding method completes on the CIM server. In
1521 this case, the requested CIM operation may not be able to successfully complete and the service may not
1522 be able to return the output for the operation. The service responds with a SOAP fault message containing
1523 the exception instances according to the following rules:

1524 **R12-1:** If a service receives a WS-Management request message that translates into a CIM intrinsic or
1525 extrinsic method, the execution of the method results in one or more exceptions, the requested CIM
1526 operation does not complete, and the service is not able to return the output for the operation, the
1527 service should respond with a SOAP fault.

1528 **R12-2:** A service responding to a WS-Management request that translated into a CIM intrinsic or
1529 extrinsic method that did not complete and resulted in an exception should include each resultant
1530 exception object as peers in the SOAP fault's Detail element. The XML representation of each
1531 exception object shall conform to the mapping rules for CIM instances defined in the [WS-CIM Mapping](#)
1532 [Specification](#).

1533 **R12-3:** A service responding to a WS-Management request that translated into a CIM intrinsic or
1534 extrinsic method that did not complete and resulted in an exception should use WS-Management fault
1535 subcodes that correspond to the nature of the exception that has occurred. If the exception does not
1536 correspond to any defined WS-Management fault subcode, the service should use the
1537 wsmb:CIMException subcode.

1538 EXAMPLE: A fault response for an extrinsic method containing an invalid method parameter that results in a CIM
 1539 exception would have the following structure:

```

1540 (1) <env:Fault>
1541 (2) <env:Code>
1542 (3)   <env:Value>env:Sender</env:Value>
1543 (4)   <env:Subcode>
1544 (5)     <env:Value>wsman:InvalidParameter</env:Value>
1545 (6)   </env:Subcode>
1546 (7) </env:Code>
1547 (8) <env:Reason>
1548 (9)   <env:Text xml:lang="en">
1549 (10)     The invocation of CIM method RequestStateChange
1550 (11)     failed because the unknown parameter Spongebob
1551 (12)     has been supplied.
1552 (13)   </env:Text>
1553 (14) </env:Reason>
1554 (15) <env:Detail>
1555 (16)   <wsman:FaultDetail>
1556 (17)     http://schemas.dmtf.org/wbem/wsman/1/wsman/faultDetail/InvalidName
1557 (18)   </wsman:FaultDetail>
1558 (19)   <cimerr:CIM_Error>
1559 (20)     ...as in WS-CIM...
1560 (21)   </cimerr:CIM_Error>
1561 (22) </env:Detail>
1562 (23) </env:Fault>

```

1563 For further information on the mapping of CIM exceptions to WS-Management fault subcodes, see
 1564 clause 17.

1565 13 CIM Specific WS-Management Options

1566 This specification relies on the WS-Management OptionSet extensibility mechanism for common scenarios.

1567 13.1 ShowExtensions Option

1568 Some of the optional CIM properties may be expensive to calculate; as a result, they are not included in
 1569 casual queries for the resource representation. Also, in some CIM Server implementations, the CIM Server
 1570 may define additional system properties that are stored along with the standard CIM properties of a given
 1571 class and that are exposed using the open content model defined in the XML Schema specified in the [WS-
 1572 CIM Mapping Specification](#).

1573 The use of ShowExtensions allows a client to indicate that the XML resource representation should contain
 1574 the elements that are expensive to calculate and the extension elements, along with the rest of the
 1575 resource properties. The ShowExtensions option may be applied to the WS-Transfer Get message, the
 1576 WS-Enumeration Enumerate message, and the WS-Eventing Subscribe message.

1577 When this option is applied to Enumerate, it communicates the desire for all resource representations
 1578 returned by the enumeration sequence to include the extensions independent of whether they are returned
 1579 in an EnumerateResponse or a PullResponse message.

1580 When this option is applied to a Subscribe message, it communicates the desire for all events matching
 1581 that Subscribe message to be returned with the extensions.

1582 This specification does not define any meaning for the ShowExtensions option on other messages. If
 1583 necessary, the client may place extra content in Put and Create messages using the extension mechanism
 1584 defined in the [WS-CIM Mapping Specification](#).

1585 Because vendor extensions can be large or expensive to retrieve, a standard option has been defined to
 1586 enable or disable the vendor extensions to be returned with the resource representation. The default is to
 1587 disable the return of vendor extensions.

1588 To show all extensions, a client sets the Option value to ShowExtensions, as follows:

```
1589 (1) <wsman:OptionSet>
1590 (2)   <wsman:Option name="ShowExtensions" />
1591 (3) </wsman:OptionSet>
```

1592 To hide extensions, a client omits or sets the Option to FALSE or 0. Any other value or an empty element
 1593 implies that the extensions should be shown.

1594 **R13.1-1:** If a service receives a request with an OptionSet containing an Option named
 1595 ShowExtensions in which the OptionSet header has mustUnderstand="TRUE" and the Option element
 1596 has mustComply="TRUE" and the value of the Option element is FALSE or 0, the service shall return
 1597 the representation in minimal form or issue a fault.

1598 **R13.1-2:** If a service receives a request with an OptionSet containing an Option named
 1599 ShowExtensions in which the OptionSet header has mustUnderstand="TRUE" and the Option element
 1600 has mustComply="TRUE" and the value of the Option element is neither false nor 0, the service shall
 1601 return the representation with additional information including the cim:Key and cim:Version attributes as
 1602 per the [WS-CIM Mapping Specification](#) and any vendor-defined extensions or issue a fault.

1603 **R13.1-3:** In the absence of this option (or mustComply requirements), a service should return the
 1604 representation in minimal form or issue a fault.

1605 **EXAMPLE:** The following shows an example representation from a service that has implemented CIM schema
 1606 version 2.11.0 that includes extensions. Note that all the vendor-specific properties come after the class
 1607 properties.

```
1608 (1) <CIM_ComputerSystem
1609 (2)   xmlns="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ComputerSystem"
1610 (3)   xmlns:cim="http://schemas.dmtf.org/wbem/wscim/1/common"
1611 (4)   xmlns:v="http://vendor.com/..."
1612 (5)   cim:Version="2.7.0">
1613 (6)
1614 (7)   <CreationClassName cim:Key="true"> ... </CreationClassName>
1615 (8)   <Name cim:Key="true"> Blue-04 </Name>
1616 (9)   <PrimaryOwnerName> Dave </PrimaryOwnerName>
1617 (10)   ...
1618 (11)   <v:PropetyCount>17</v:PropertyCount>
1619 (12) </CIM_ComputerSystem>
```

1620 14 Instance Representation

1621 Instances are represented according to the XML namespace defined by the [WS-CIM Mapping](#)
 1622 [Specification](#). This clause defines additional constraints on that representation.

1623 WS-CIM allows references to be represented using a variety of addressing models; however, this
 1624 specification is associated with WS-Management, which uses a specific addressing model.

1625 **R14-1:** A service shall accept and return only instance representations in which XML elements
 1626 corresponding to CIM reference properties are represented as a WS-Addressing EPR using the
 1627 wsa XML namespace defined in clause 6.

1628 **15 Fault Codes**

1629 Faults defined in this specification must use the following action URI:

1630 `http://schemas.dmtf.org/wbem/wsman/1/cimbinding/fault`

1631 **15.1 wsmb:CIMException**

1632 Table 6 provides information about the wsmb:CIMException fault subcode.

1633 **Table 6 – wsmb:CIMException**

Fault Subcode	wsmb:CIMException
Action URI	<code>http://schemas.dmtf.org/wbem/wsman/1/cimbinding/fault</code>
Code	s:Receiver
Reason	The CIM server encountered an exception during the processing of the request.
Detail	XML representation of CIM_Error instance
Comments	
Applicability	Any message
Remedy	Depends upon the exception

1634 **15.2 wsmb:PolymorphismModeNotSupported**

1635 Table 7 provides information about the wsmb:PolymorphismModeNotSupported fault subcode.

1636 **Table 7 – wsmb:PolymorphismModeNotSupported**

Fault Subcode	wsmb:PolymorphismModeNotSupported
Action URI	<code>http://schemas.dmtf.org/wbem/wsman/1/cimbinding/fault</code>
Code	s:Sender
Reason	The resource does not support the requested polymorphism mode.
Detail	
Comments	
Applicability	wse:Enumerate, wse:Subscribe
Remedy	Try the request again without specifying a polymorphism mode.

1637 **16 Mapping for DSP0200 CIM Operations**

1638 CIM profiles define support for CIM operations for each CIM class used in the profiles. These supported
 1639 operations are defined in [DSP0200](#). This clause outlines the WS-Management equivalent operations for
 1640 each supported CIM operation that is defined in [DSP0200](#) and additional uses of WS-Management
 1641 functionality to achieve the same goal.

1642 **16.1 Supported Operations**

1643 The following CIM operations have equivalents defined by this specification:

- 1644 • **GetInstance:** This operation is used to return a single CIM instance from the target namespace.
- 1645 • **DeleteInstance:** This operation is used to delete a single CIM instance from the target
 1646 namespace.
- 1647 • **ModifyInstance:** This operation is used to modify a single CIM instance in the target namespace.
- 1648 • **CreateInstance:** This operation is used to create a single CIM instance in the target namespace.
- 1649 • **EnumerateInstances:** This operation is used to enumerate instances of a CIM Class (this includes
 1650 instances in the class and any subclasses in accordance with the polymorphic nature of CIM
 1651 objects) in the target Namespace.
- 1652 • **EnumerateInstanceNames:** This operation is used to enumerate the names (model paths) of the
 1653 instances of a CIM Class (this includes instances in the class and any subclasses in accordance
 1654 with the polymorphic nature of CIM objects) in the target Namespace.
- 1655 • **Associators:** This operation is used to enumerate CIM Objects (Classes or Instances) that are
 1656 associated to a particular source CIM Object.
- 1657 • **AssociatorNames:** This operation is used to enumerate the names of CIM Objects (Classes or
 1658 Instances) that are associated to a particular source CIM Object.
- 1659 • **References:** This operation is used to enumerate the association objects that refer to a particular
 1660 target CIM Object (Class or Instance).
- 1661 • **ReferenceNames:** This operation is used to enumerate the association objects that refer to a
 1662 particular target CIM Object (Class or Instance).
- 1663 • **OpenEnumerateInstances:** The OpenEnumerateInstances operation establishes and opens an
 1664 enumeration session of the instances of a CIM class (including instances of its subclasses) in the
 1665 target namespace. Optionally, it retrieves a first set of instances.
- 1666 • **OpenEnumerateInstancePaths:** The OpenEnumerateInstancePaths operation establishes and
 1667 opens an enumeration session of the instance paths of the instances of a CIM class (including
 1668 instances of its subclasses) in the target namespace. Optionally, it retrieves a first set of instance
 1669 paths.
- 1670 • **OpenReferenceInstances:** The OpenReferenceInstances operation establishes and opens the
 1671 enumeration session of association instances that refer to a particular target CIM instance in the
 1672 target namespace. Optionally, it retrieves a first set of instances.
- 1673 • **OpenReferenceInstancePaths:** The OpenReferenceInstancePaths operation establishes and
 1674 opens an enumeration session of the instance paths of the association instances that refer to a
 1675 particular target CIM instance in the target namespace. Optionally, it retrieves a first set of
 1676 instance paths.
- 1677 • **OpenAssociatorInstances:** The OpenAssociatorInstances operation establishes and opens an
 1678 enumeration session of the instances associated with a particular source CIM instance in the
 1679 target namespace. Optionally, it retrieves a first set of instances.

- 1680 • OpenAssociatorInstancePaths: The OpenAssociatorInstancePaths operation establishes and
1681 opens an enumeration session of the instance paths of the instances associated with a particular
1682 source CIM instance in the target namespace. Optionally, it retrieves a first set of instance paths.
- 1683 • PullInstancesWithPath: The PullInstancesWithPath operation retrieves instances including their
1684 instance paths from an open enumeration session represented by an enumeration context value.
- 1685 • PullInstancePaths: The PullInstancePaths operation retrieves instance paths from an open
1686 enumeration session represented by an enumeration context value.
- 1687 • CloseEnumeration: The CloseEnumeration operation closes an open enumeration session,
1688 performing an early termination of an enumeration sequence.

1689 The following sub-sections define the mapping of the above operations over WS-Management.

1690 **16.1.1 GetInstance**

1691 The mapping defined in Table 8 shall be used for the GetInstance operation.

1692 **Table 8 – GetInstance**

Operation	GetInstance
Operation target	CIM namespace
WS-Man operation	WS-Transfer:Get
WS-Man EPR	Class-specific ResourceURI with keys as selectors
Additional usage	None
Notes	As defined in R7.1-1, the class specified in the Resource URI needs to be the creation class of the instance it addresses.

1693 Table 9 provides the mapping of GetInstance arguments defined in Section 2.3.2.2 of [DSP0200](#).

1694 **Table 9 – GetInstance Arguments**

Argument	GetInstance
InstanceName	Mapped to WS-Man EPR
LocalOnly	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
IncludeQualifier	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Transfer Get. If it is not NULL, then the operation is handled through fragment level WS-Transfer Get (see Section 4.9 of DSP0226).

1695 Table 10 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1696 [DSP0226](#).

1697 **Table 10 – GetInstance Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FAILED	wsman:InternalError

1698 **16.1.2 DeleteInstance**

1699 The mapping defined in Table 11 shall be used for the DeleteInstance operation.

1700 **Table 11 – DeleteInstance**

Operation	DeleteInstance
Operation target	CIM namespace
WS-Man operation	WS-Transfer:Delete or WS-Eventing:Unsubscribe (for CIM_IndicationSubscription and CIM_FilterCollectionSubscription)
WS-Man EPR	Class-specific ResourceURI with keys as selectors
Additional usage	None

1701 Table 12 provides the mapping of the DeleteInstance arguments defined in Section 2.3.2.4 of [DSP0200](#).

1702 **Table 12 – DeleteInstance Arguments**

Argument	DeleteInstance
InstanceName	Mapped to WS-Man EPR

1703 Table 13 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1704 [DSP0226](#).

1705 **Table 13 – DeleteInstance Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable

Status Code	Equivalent SOAP Fault
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FAILED	wsman:InternalError

1706 **16.1.3 ModifyInstance**

1707 The mapping defined in Table 14 shall be used for the ModifyInstance operation.

1708 **Table 14 – ModifyInstance**

Operation	ModifyInstance
Operation target	CIM namespace
WS-Man operation	WS-Transfer:Put or WS-Eventing:Renew (for CIM_IndicationSubscription and CIM_FilterCollectionSubscription)
WS-Man EPR	Class-specific ResourceURI with keys as selectors
Additional usage	None
Notes	As defined in R7.1-1, the class specified in the Resource URI needs to be the creation class of the instance it addresses.

1709 Table 15 provides the mapping of the ModifyInstance arguments defined in Section 2.3.2.8 of [DSP0200](#).

1710 **Table 15 – ModifyInstance Arguments**

Argument	ModifyInstance
InstanceName	Mapped to WS-Man EPR
IncludeQualifier	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	Always set to NULL for the instances of CIM_IndicationSubscription and CIM_FilterCollectionSubscription. For instances of other classes: If it is NULL, then the operation is handled through WS-Transfer Put. If it is not NULL, then the operation is handled through fragment level WS-Transfer Put (see Section 4.10 of DSP0226).

1711 Table 16 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
1712 [DSP0226](#).

1713 **Table 16 – ModifyInstance Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported

Status Code	Equivalent SOAP Fault
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FAILED	wsman:InternalError

1714 **16.1.4 CreateInstance**

1715 The mapping defined in Table 17 shall be used for the CreateInstance operation.

1716 **Table 17 – CreateInstance**

Operation	CreateInstance
Operation target	CIM namespace
WS-Man operation	WS-Transfer:Create or WS-Eventing:Subscribe (for CIM_IndicationSubscription and CIM_FilterCollectionSubscription)
WS-Man EPR	Class-specific ResourceURI as factory, with only the __cimnamespace selector allowed
Additional usage	None
Notes	As defined in R7.1-1, the class specified in the Resource URI needs to be the creation class of the instance it addresses.

1717 Table 18 provides the mapping of the CreateInstance arguments as defined in Section 2.3.2.6 of [DSP0200](#).

1718 **Table 18 – CreateInstance Arguments**

Argument	CreateInstance
InstanceName	Mapped to WS-Man EPR

1719 Table 19 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in [DSP0226](#).

1721 **Table 19 – CreateInstance Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable

Status Code	Equivalent SOAP Fault
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_ALREADY_EXISTS	wsman:AlreadyExists
CIM_ERR_FAILED	wsman:InternalError

1722 **16.1.5 EnumerateInstances**

1723 The mapping defined in Table 20 shall be used for the EnumerateInstances operation.

1724 **Table 20 – EnumerateInstances**

Operation	EnumerateInstances
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate and WS-Enumeration:Pull (if needed)
WS-Man EPR	Class-specific ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateObjectAndEPR
Notes	

1725 Table 21 provides the mapping of EnumerateInstances arguments as defined in Section 2.3.2.11 of
 1726 [DSP0200](#).

1727 **Table 21 – EnumerateInstances Arguments**

Argument	EnumerateInstances
ClassName	Mapped to WS-Man EPR
LocalOnly	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
DeepInheritance	If true, then wsmb:PolymorphismMode modifier element value is set to IncludeSubClassProperties or wsmb:PolymorphismMode is not specified. If false, then wsmb:PolymorphismMode modifier element value is set to ExcludeSubClassProperties.
IncludeQualifier	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Enumeration. If it is not NULL, then the operation is handled through fragment-level enumerations (see Section 5.6 of DSP0226).

1728 Table 22 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1729 [DSP0226](#).

1730 **Table 22 – EnumerateInstances Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_FAILED	wsman:InternalError

1731 **16.1.6 EnumerateInstanceNames**

1732 The mapping defined in Table 23 shall be used for the EnumerateInstanceNames operation.

1733 **Table 23 – EnumerateInstanceNames**

Operation	EnumerateInstanceNames
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate and WS-Enumeration:Pull (if needed)
WS-Man EPR	Class-specific ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateEPR
Notes	

1734 Table 24 provides the mapping of EnumerateInstanceNames arguments as defined in Section 2.3.2.12 of
 1735 [DSP0200](#).

1736 **Table 24 – EnumerateInstanceNames Arguments**

Argument	EnumerateInstanceNames
ClassName	Mapped to WS-Man EPR

1737 Table 25 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1738 [DSP0226](#).

1739 **Table 25 – EnumerateInstanceNames Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_FAILED	wsman:InternalError

1740 **16.1.7 Associators**

1741 The mapping defined in Table 26 shall be used for the Associators operation.

1742 **Table 26 – Associators**

Operation	Associators
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate and WS-Enumeration:Pull (if needed)
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateObjectAndEPR Use the following association filter dialect with the wsmb:AssociatedInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1743 Table 27 provides the mapping of the Associators arguments as defined in Section 2.3.2.14 of [DSP0200](#).

1744 **Table 27 – Associators Arguments**

Argument	Associators
ObjectName	wsmb:Object value is set to ObjectName
AssocClass	If not NULL, wsmb:AssociationClassName value is set to AssocClass
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
ResultRole	If not NULL, wsmb:ResultRole value is set to ResultRole
IncludeQualifiers	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.

Argument	Associators
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Enumeration. If it is not NULL, then the operation is handled through fragment-level enumerations (see Section 5.6 of DSP0226).

1745 Table 28 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1746 [DSP0226](#).

1747 **Table 28 – Associators Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_FAILED	wsman:InternalError

1748 **16.1.8 AssociatorNames**

1749 The mapping defined in Table 29 shall be used for the AssociatorNames operation.

1750 **Table 29 – AssociatorNames**

Operation	AssociatorNames
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate and WS-Enumeration:Pull (if needed)
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateEPR Use the following association filter dialect with the wsmb:AssociatedInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1751 Table 30 provides the mapping of the AssociatorNames arguments as defined in Section 2.3.2.15 of
 1752 [DSP0200](#).

1753 **Table 30 – AssociatorNames Arguments**

Argument	AssociatorNames
ObjectName	wsmb:Object value is set to ObjectName
AssocClass	If not NULL, wsmb:AssociationClassName value is set to AssocClass
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
ResultRole	If not NULL, wsmb:ResultRole value is set to ResultRole

1754 Table 31 provides the mapping of status codes as defined in [DSP0200](#) to equivalent SOAP faults defined
 1755 in [DSP0226](#).

1756 **Table 31 – AssociatorNames Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_FAILED	wsman:InternalError

1757 16.1.9 References

1758 The mapping defined in Table 32 shall be used for the References operation.

1759 **Table 32 – References**

Operation	References
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate and WS-Enumeration:Pull (if needed)
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateObjectAndEPR Use association the following filter dialect with the wsmb:AssociationInstances element: http://schemas.dmf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1760 Table 33 provides the mapping of the References arguments as defined in Section 2.3.2.16 of [DSP0200](#).

1761 **Table 33 – References Arguments**

Argument	References
ObjectName	wsmb:Object value is set to ObjectName
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
IncludeQualifiers	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Enumeration. If it is not NULL, then the operation is handled through fragment-level enumerations (see Section 5.6 of DSP0226).

1762 Table 34 provides the mapping of status codes as defined in [DSP0200](#) to equivalent SOAP faults defined
1763 in [DSP0226](#).

1764 **Table 34 – References Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_FAILED	wsman:InternalError

1765 **16.1.10 ReferenceNames**

1766 The mapping defined in Table 35 shall be used for the ReferenceNames operation.

1767 **Table 35 – ReferenceNames**

Operation	ReferenceNames
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate and WS-Enumeration:Pull (if needed)
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateEPR Use association the following filter dialect with the wsmb:AssociationInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1768 Table 36 provides the mapping of the ReferenceNames arguments as defined in Section 2.3.2.17 of
 1769 [DSP0200](#).

1770 **Table 36 – ReferenceNames Arguments**

Argument	ReferenceNames
ObjectName	wsmb:Object value is set to ObjectName
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role

1771 Table 37 provides the mapping of status codes as defined in [DSP0200](#) to equivalent SOAP faults defined
 1772 in [DSP0226](#).

1773 **Table 37 – ReferenceNames Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_NOT_SUPPORTED (by the CIM Server for this operation)	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_SUPPORTED (This operation is not supported for the class of the specified instance, if provided.)	wsa:ActionNotSupported
CIM_ERR_FAILED	wsman:InternalError

1774 **16.1.11 Pulled Enumerations**

1775 The following operations used in pulled enumerations are mapped in this section.

- 1776 1) Open operations
 - 1777 a) OpenEnumerateInstances
 - 1778 b) OpenEnumerateInstancePaths
 - 1779 c) OpenReferenceInstances
 - 1780 d) OpenReferenceInstancePaths
 - 1781 e) OpenAssociatorInstances
 - 1782 f) OpenAssociatorInstancePaths
- 1783 2) Pull operations
 - 1784 a) PullInstancesWithPath
 - 1785 b) PullInstancePaths
- 1786 3) Other
 - 1787 a) CloseEnumeration

1788 **16.1.11.1 OpenEnumerateInstances**

1789 The mapping defined in Table 38 shall be used for the OpenEnumerateInstances operation.

1790 **Table 38 – OpenEnumerateInstances**

Operation	OpenEnumerateInstances
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate
WS-Man EPR	Class-specific ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateObjectAndEPR
Notes	

1791 Table 39 provides the mapping of OpenEnumerateInstances arguments as defined in Section 5.3.2.24.3 of
 1792 [DSP0200](#).

1793 **Table 39 – OpenEnumerateInstances Arguments**

Argument	OpenEnumerateInstances
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
ClassName	Mapped to WS-Man EPR
DeepInheritance	If true, then wsmb:PolymorphismMode modifier element value is set to IncludeSubClassProperties or wsmb:PolymorphismMode is not specified. If false, then wsmb:PolymorphismMode modifier element value is set to ExcludeSubClassProperties.
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Enumeration. If it is not NULL, then the operation is handled through fragment-level enumerations (see Section 5.6 of DSP0226).
FilterQueryLanguage	See section 8.3 of DSP0226
FilterQuery	See section 8.3 of DSP0226
OperationTimeOut	Mapped to wsen:Expires
ContinueOnError	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
MaxObjectCount	Mapped to wsman:MaxElements

1794 Table 40 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1795 [DSP0226](#).

1796 **Table 40 – OpenEnumerateInstances Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable

Status Code	Equivalent SOAP Fault
CIM_ERR_INVALID_OPERATION_TIMEOUT	wsen:InvalidExpirationTime
CIM_ERR_CONTINUATION_ON_ERROR_NOT_SUPPORTED	Not applicable.
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable
CIM_ERR_FILTERED_ENUMERATION_NOT_SUPPORTED	wsen:FilteringNotSupported
CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	wsen:FilterDialectRequestedUnavailable
CIM_ERR_INVALID_QUERY	wsen:CannotProcessFilter
CIM_ERR_FAILED	wsman:InternalError

1797 **16.1.11.2 OpenEnumerateInstancePaths**

1798 The mapping defined in Table 41 shall be used for the OpenEnumerateInstancePaths operation.

1799 **Table 41 – OpenEnumerateInstancePaths**

Operation	OpenEnumerateInstancePaths
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate
WS-Man EPR	Class-specific ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateEPR
Notes	

1800 Table 42 provides the mapping of OpenEnumerateInstancePaths arguments as defined in Section
 1801 5.3.2.24.4 of [DSP0200](#).

1802 **Table 42 – OpenEnumerateInstancePaths Arguments**

Argument	OpenEnumerateInstancePaths
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
ClassName	Mapped to WS-Man EPR
FilterQueryLanguage	See section 8.3 of DSP0226
FilterQuery	See section 8.3 of DSP0226
OperationTimeOut	Mapped to wsen:Expires
ContinueOnError	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
MaxObjectCount	Mapped to wsman:MaxElements

1803 Table 43 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1804 [DSP0226](#).

1805 **Table 43 – OpenEnumerateInstancePaths Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_OPERATION_TIMEOUT	wsen:InvalidExpirationTime
CIM_ERR_CONTINUATION_ON_ERROR_NOT_SUPPORTED	Not applicable.
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_CLASS	wsa:DestinationUnreachable
CIM_ERR_FILTERED_ENUMERATION_NOT_SUPPORTED	wsen:FilteringNotSupported
CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	wsen:FilterDialectRequestedUnavailable
CIM_ERR_INVALID_QUERY	wsen:CannotProcessFilter
CIM_ERR_FAILED	wsman:InternalError

1806 **16.1.11.3 OpenReferenceInstances**

1807 The mapping defined in Table 44 shall be used for the OpenReferenceInstances operation.

1808 **Table 44 – OpenReferenceInstances**

Operation	OpenReferenceInstances
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateObjectAndEPR Use association the following filter dialect with the wsmb:AssociationInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1809 Table 45 provides the mapping of OpenReferenceInstances arguments as defined in Section 5.3.2.24.5 of
 1810 [DSP0200](#).

1811 **Table 45 – OpenReferenceInstances Arguments**

Argument	OpenReferenceInstances
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
InstanceName	wsmb:Object value is set to InstanceName

Argument	OpenReferenceInstances
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Enumeration. If it is not NULL, then the operation is handled through fragment-level enumerations (see Section 5.6 of DSP0226).
FilterQueryLanguage	See section 8.3 of DSP0226
FilterQuery	See section 8.3 of DSP0226
OperationTimeOut	Mapped to wsen:Expires
ContinueOnError	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
MaxObjectCount	Mapped to wsman:MaxElements

1812 Table 46 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1813 [DSP0226](#).

1814 **Table 46 – OpenReferenceInstances Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_OPERATION_TIMEOUT	wsen:InvalidExpirationTime
CIM_ERR_CONTINUATION_ON_ERROR_NOT_SUPPORTED	Not applicable.
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FILTERED_ENUMERATION_NOT_SUPPORTED	wsen:FilteringNotSupported
CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	wsen:FilterDialectRequestedUnavailable
CIM_ERR_INVALID_QUERY	wsen:CannotProcessFilter
CIM_ERR_FAILED	wsman:InternalError

1815 **16.1.11.4 OpenReferenceInstancePaths**

1816 The mapping defined in Table 47 shall be used for the OpenReferenceInstancePaths operation.

1817 **Table 47 – OpenReferenceInstancePaths**

Operation	OpenReferenceInstancePaths
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate

Operation	OpenReferenceInstancePaths
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateEPR Use association the following filter dialect with the wsmb:AssociationInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1818 Table 48 provides the mapping of OpenReferenceInstancePaths arguments as defined in Section
 1819 5.3.2.24.6 of [DSP0200](#).

1820 **Table 48 – OpenReferenceInstancePaths Arguments**

Argument	OpenReferenceInstancePaths
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
InstanceName	wsmb:Object value is set to InstanceName
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
FilterQueryLanguage	See section 8.3 of DSP0226
FilterQuery	See section 8.3 of DSP0226
OperationTimeOut	Mapped to wsen:Expires
ContinueOnError	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
MaxObjectCount	Mapped to wsman:MaxElements

1821 Table 49 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1822 [DSP0226](#).

1823 **Table 49 – OpenReferenceInstancePaths Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_OPERATION_TIMEOUT	wsen:InvalidExpirationTime
CIM_ERR_CONTINUATION_ON_ERROR_NOT_SUPPORTED	Not applicable.
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FILTERED_ENUMERATION_NOT_SUPPORTED	wsen:FilteringNotSupported
CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	wsen:FilterDialectRequestedUnavailable

Status Code	Equivalent SOAP Fault
CIM_ERR_INVALID_QUERY	wsen:CannotProcessFilter
CIM_ERR_FAILED	wsman:InternalError

1824 **16.1.11.5 OpenAssociatorInstances**

1825 The mapping defined in Table 50 shall be used for the OpenAssociatorInstances operation.

1826 **Table 50 – OpenAssociatorInstances**

Operation	OpenAssociatorInstances
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateObjectAndEPR Use the following association filter dialect with the wsmb:AssociatedInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1827 Table 51 provides the mapping of OpenAssociatorInstances arguments as defined in Section 5.3.2.24.7 of
1828 [DSP0200](#).1829 **Table 51 – OpenAssociatorInstances Arguments**

Argument	OpenAssociatorInstances
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
InstanceName	wsmb:Object value is set to InstanceName
AssocClass	If not NULL, wsmb:AssociationClassName value is set to AssocClass
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
ResultRole	If not NULL, wsmb:ResultRole value is set to ResultRole
IncludeClassOrigin	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
PropertyList[]	If it is NULL, then the operation is handled through WS-Enumeration. If it is not NULL, then the operation is handled through fragment-level enumerations (see Section 5.6 of DSP0226).
FilterQueryLanguage	See section 8.3 of DSP0226
FilterQuery	See section 8.3 of DSP0226
OperationTimeout	Mapped to wsen:Expires
ContinueOnError	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
MaxObjectCount	Mapped to wsman:MaxElements

1830 Table 52 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1831 [DSP0226](#).

1832 **Table 52 – OpenAssociatorInstances Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_OPERATION_TIMEOUT	wsen:InvalidExpirationTime
CIM_ERR_CONTINUATION_ON_ERROR_NOT_SUPPORTED	Not applicable.
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FILTERED_ENUMERATION_NOT_SUPPORTED	wsen:FilteringNotSupported
CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	wsen:FilterDialectRequestedUnavailable
CIM_ERR_INVALID_QUERY	wsen:CannotProcessFilter
CIM_ERR_FAILED	wsman:InternalError

1833 **16.1.11.6 OpenAssociatorInstancePaths**

1834 The mapping defined in Table 53 shall be used for the OpenAssociatorInstancePaths operation.

1835 **Table 53 – OpenAssociatorInstancePaths**

Operation	OpenAssociatorInstancePaths
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Enumerate
WS-Man EPR	All-classes ResourceURI with no selectors
Additional usage	Use wsman:EnumerationMode=EnumerateEPR Use the following association filter dialect with the wsmb:AssociatedInstances element: http://schemas.dmtf.org/wbem/wsman/1/cimbinding/associationFilter
Notes	

1836 Table 54 provides the mapping of OpenAssociatorInstancePaths arguments as defined in Section
 1837 5.3.2.24.8 of [DSP0200](#).

1838 **Table 54 – OpenAssociatorInstancePaths Arguments**

Argument	OpenAssociatorInstancePaths
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
InstanceName	wsmb:Object value is set to InstanceName

Argument	OpenAssociatorInstancePaths
AssocClass	If not NULL, wsmb:AssociationClassName value is set to AssocClass
ResultClass	If not NULL, wsmb:ResultClassName value is set to ResultClass
Role	If not NULL, wsmb:Role value is set to Role
ResultRole	If not NULL, wsmb:ResultRole value is set to ResultRole
FilterQueryLanguage	See section 8.3 of DSP0226
FilterQuery	See section 8.3 of DSP0226
OperationTimeout	Mapped to wsen:Expires
ContinueOnError	There is no corresponding WS-Management parameter for this argument. From the expected behavior perspective, this argument is treated as always set to false.
MaxObjectCount	Mapped to wsman:MaxElements

1839 Table 55 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1840 [DSP0226](#).

1841 **Table 55 – OpenAssociatorInstancePaths Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_OPERATION_TIMEOUT	wsen:InvalidExpirationTime
CIM_ERR_CONTINUATION_ON_ERROR_NOT_SUPPORTED	Not applicable.
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_NOT_FOUND	wsa:DestinationUnreachable
CIM_ERR_FILTERED_ENUMERATION_NOT_SUPPORTED	wsen:FilteringNotSupported
CIM_ERR_QUERY_LANGUAGE_NOT_SUPPORTED	wsen:FilterDialectRequestedUnavailable
CIM_ERR_INVALID_QUERY	wsen:CannotProcessFilter
CIM_ERR_FAILED	wsman:InternalError

1842 **16.1.11.7 PullInstancesWithPath**

1843 The mapping defined in Table 56 shall be used for the PullInstancesWithPath operation.

1844 **Table 56 – PullInstancesWithPath**

Operation	PullInstancesWithPath
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Pull
WS-Man EPR	Class-specific ResourceURI with no selectors or All-classes ResourceURI with no selectors as specified in the corresponding WS-Enumeration:Enumerate operation

Operation	PullInstancesWithPath
Notes	The corresponding WS-Enumerate:Enumerate operation shall specify wsman:EnumerationMode=EnumerateObjectAndEPR and the association filter dialect with the appropriate element (if needed).

1845 Table 57 provides the mapping of PullInstancesWithPath arguments as defined in Section 5.3.2.24.10 of
 1846 [DSP0200](#).

1847 **Table 57 – PullInstancesWithPath Arguments**

Argument	PullInstancesWithPath
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
MaxObjectCount	Mapped to wsman:MaxElements

1848 Table 58 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1849 [DSP0226](#).

1850 **Table 58 – PullInstancesWithPath Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_ENUMERATION_CONTEXT	wsen:InvalidEnumerationContext
CIM_ERR_PULL_HAS_BEEN_ABANDONED	wsman:InternalError
CIM_ERR_SERVER_LIMITS_EXCEEDED	wsman:InternalError
CIM_ERR_FAILED	wsman:InternalError

1851 **16.1.11.8 PullInstancePaths**

1852 The mapping defined in Table 59 shall be used for the PullInstancePaths operation.

1853 **Table 59 – PullInstancePaths**

Operation	PullInstancePaths
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Pull
WS-Man EPR	Class-specific ResourceURI with no selectors or All-classes ResourceURI with no selectors as specified in the corresponding WS-Enumeration:Enumerate operation
Notes	The corresponding WS-Enumerate:Enumerate operation shall specify wsman:EnumerationMode=EnumerateEPR and the association filter dialect with the appropriate element (if needed).

1854 Table 60 provides the mapping of PullInstancePaths arguments as defined in Section 5.3.2.24.11 of
 1855 [DSP0200](#).

1856 **Table 60 – PullInstancePaths Arguments**

Argument	PullInstancePaths
EnumerationContext	Mapped to wsen:EnumerationContext
EndOfSequence	Mapped to wsman:EndOfSequence
MaxObjectCount	Mapped to wsman:MaxElements

1857 Table 61 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1858 [DSP0226](#).

1859 **Table 61 – PullInstancePaths Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_ENUMERATION_CONTEXT	wsen:InvalidEnumerationContext
CIM_ERR_PULL_HAS_BEEN_ABANDONED	wsman:InternalError
CIM_ERR_SERVER_LIMITS_EXCEEDED	wsman:InternalError
CIM_ERR_FAILED	wsman:InternalError

1860 **16.1.11.9 CloseEnumeration**

1861 The mapping defined in Table 62 shall be used for the CloseEnumeration operation.

1862 **Table 62 – CloseEnumeration**

Operation	CloseEnumeration
Operation target	CIM namespace
WS-Man operation	WS-Enumeration:Release

1863 Table 63 provides the mapping of CloseEnumeration arguments as defined in Section 5.3.2.24.12 of
 1864 [DSP0200](#).

1865 **Table 63 – CloseEnumeration Arguments**

Argument	CloseEnumeration
EnumerationContext	Mapped to wsen:EnumerationContext

1866 Table 64 provides the mapping of status codes defined in [DSP0200](#) to equivalent SOAP faults defined in
 1867 [DSP0226](#).

1868 **Table 64 – CloseEnumeration Error Codes**

Status Code	Equivalent SOAP Fault
CIM_ERR_ACCESS_DENIED	wsman:AccessDenied
CIM_ERR_SERVER_IS_SHUTTING_DOWN	wsman:InternalError
CIM_ERR_NOT_SUPPORTED	wsa:ActionNotSupported
CIM_ERR_INVALID_NAMESPACE	wsa:DestinationUnreachable
CIM_ERR_INVALID_PARAMETER	wsman:InvalidParameter
CIM_ERR_INVALID_ENUMERATION_CONTEXT	wsen:InvalidEnumerationContext
CIM_ERR_PULL_CANNOT_BE_ABANDONED	wsman:InternalError
CIM_ERR_FAILED	wsman:InternalError

1869 **16.1.12 ExecQuery**

1870 This operation is supported for the CQL query language. See 8.1 for more details.

1871 **16.2 Unsupported Operations**

1872 This specification does not define equivalents for the following operations:

- 1873 • GetClass
- 1874 • DeleteClass
- 1875 • CreateClass
- 1876 • ModifyClass
- 1877 • EnumerateClasses
- 1878 • EnumerateClassNames
- 1879 • GetProperty
- 1880 • SetProperty
- 1881 • GetQualifier
- 1882 • SetQualifier
- 1883 • DeleteQualifier
- 1884 • EnumerateQualifiers
- 1885 • OpenQueryInstances
- 1886 • PullInstances
- 1887 • EnumerationCount

1888 **17 Mapping of Error Messages to SOAP Fault Subcodes**

1889 Table 65 outlines suggested mappings of CIM error messages to corresponding subcodes to be used when
 1890 returning SOAP faults.

1891 **Table 65 – CIM Error Messages with Corresponding Subcode Mappings**

Message ID	Message Name	Fault Subcode
WIPG0201	Authentication failed	wsman:AccessDenied (Support may be transport-dependent.)
WIPG0202	Authorization failed	wsman:AccessDenied
WIPG0203	Operation not supported by CIM service infrastructure	wsa:ActionNotSupported
WIPG0204	CIM namespace not found	wsa:DestinationUnreachable
WIPG0205	Missing input parameter	wsmb:CIMException
WIPG0206	Duplicate input parameter	wsman:InvalidParameter
WIPG0207	Unknown input parameter	wsman:InvalidParameter
WIPG0208	Invalid input parameter value	wsman:InvalidParameter
WIPG0213	CIM instance not found	wsa:DestinationUnreachable
WIPG0214	CIM class not found	wsa:DestinationUnreachable
WIPG0216	CIM instance already exists	wsman:AlreadyExists
WIPG0218	No such CIM method	wsa:ActionNotSupported
WIPG0219	CIM method not supported by CIM class implementation	wsa:ActionNotSupported
WIPG0220	No such CIM property	wxf:InvalidRepresentation
WIPG0221	Unknown query language	wsen:FilterDialectRequestedUnavailable (if encountered while processing wsen:Enumerate) wsman:CannotProcessFilter (if encountered while processing wse:Subscribe)
WIPG0222	Query language feature not supported by WBEM service infrastructure	wsen:CannotProcessFilter (if encountered while processing wsen:Enumerate) wsman:CannotProcessFilter (for exceptions encountered while processing wse:Subscribe)
WIPG0223	Invalid query	wsen:CannotProcessFilter (if encountered while processing wsen:Enumerate) wsman:CannotProcessFilter (if encountered while processing wsen:Enumerate)
WIPG0227	Operation failure	wsman:InternalError
WIPG0228	Operation not supported by CIM class implementation	wsa:ActionNotSupported
WIPG0229	CIM method invocation not supported by WBEM service infrastructure	wsa:ActionNotSupported

1892 18 XSD

1893 A normative copy of the XML schemas ([XML Schema Part 1](#), [XML Schema Part 2](#)) for this specification
1894 may be retrieved by resolving the XML namespace URIs for this specification (listed in clause 5).

1895 19 WSDL

1896 This specification does not define a normative WSDL document. While it is possible to define a generic
1897 WSDL document that can apply to all CIM classes, it does a disservice to developers who can provide a
1898 more specific WSDL document tailored to a specific CIM class.

1899 **R19-1:** WSDL documents for a CIM class should include all WS-Transfer operations.

1900 **R19-2:** WSDL documents for a CIM class or the query engine should include all WS-Enumeration
1901 operations.

1902 **R19-3:** WSDL documents for a CIM class or the query engine should include all WS-Eventing
1903 operations.

1904 **R19-4:** WSDL documents for a CIM class should include operations for all extrinsic methods defined
1905 by the class.

1906
1907
1908
1909

ANNEX A (informative)

Change Log

Version	Date	Author	Description
1.0.0	2009-06-19	Rick Landau	DMTF Standard Release
1.0.0	2008-03-03	Hemal Shah	Draft Standard Release.

1910
1911
1912