

1
2
3
4
5



Document Number: DSP1054

Date: 2011-10-26

Version: 1.2.1

6 **Indications Profile**

7 **Document Type: Specification**

8 **Document Status: DMTF Standard**

9 **Document Language: en-US**

10

11 Copyright notice

12 Copyright © 2007, 2010, 2011 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

CONTENTS

34	Foreword	7
35	Introduction	7
36	Document conventions.....	8
37	Typographical conventions	8
38	ABNF usage conventions	8
39	Deprecated material.....	8
40	Experimental material	8
41	1 Scope	11
42	2 Normative references.....	11
43	3 Terms and definitions.....	12
44	4 Symbols and abbreviated terms.....	16
45	5 Synopsis.....	16
46	6 Description	19
47	6.1 Events and indications.....	19
48	6.1.1 Events	19
49	6.1.2 Indications.....	19
50	6.1.3 Definition of events and indications in referencing profiles	21
51	6.1.4 Indication generation, indication filtering, and indication delivery.....	21
52	6.1.5 Reliable indication delivery	23
53	6.1.6 Avoidance of repeated indication delivery	23
54	6.2 Indication filters.....	24
55	6.2.1 General	24
56	6.2.2 Indication filter coverage.....	25
57	6.2.3 Static indication filters	26
58	6.2.4 Indication-specific indication filters	26
59	6.2.5 Global indication filters.....	27
60	6.2.6 Dynamic indication filters	27
61	6.3 Filter collections	27
62	6.3.1 General	27
63	6.3.2 Filter collection coverage.....	27
64	6.3.3 Static filter collections	28
65	6.4 Subscriptions, listeners, and listener destinations.....	28
66	6.4.1 Subscriptions	28
67	6.4.2 Overlapping coverages of subscriptions.....	29
68	6.4.3 Subscription management authorization	29
69	6.4.4 Listeners	29
70	6.4.5 Listener destinations.....	29
71	6.5 Indication service and implementation.....	30
72	6.5.1 Implementation	30
73	6.5.2 Indication service	30
74	6.6 Indication system and referencing profiles	30
75	6.7 CIM model.....	31
76	7 Implementation.....	34
77	7.1 Separation of requirements	34
78	7.2 Features.....	34
79	7.2.1 DynamicIndicationFilters.....	34
80	7.2.2 IndicationServiceInitialSettingsExposed.....	35
81	7.2.3 IndicationServiceModification	35
82	7.2.4 ReliableIndications.....	35
83	7.2.5 SuppressRepeatNotificationPolicy.....	36
84	7.2.6 DelayRepeatNotificationPolicy.....	36

85	7.2.7	IndividualFilterSubscription	36
86	7.2.8	FilterCollectionCoverageExposure	36
87	7.3	Adaptations	37
88	7.3.1	Conventions	37
89	7.3.2	IndicationService: CIM_IndicationService	37
90	7.3.3	IndicationSystem: CIM_System.....	41
91	7.3.4	HostedIndicationService: CIM_HostedService.....	41
92	7.3.5	IndicationsProfileRegistration: CIM_RegisteredProfile.....	42
93	7.3.6	ElementConformsToProfile: CIM_ElementConformsToProfile	42
94	7.3.7	IndicationServiceCapabilities: CIM_IndicationServiceCapabilities.....	43
95	7.3.8	CapabilitiesOfIndicationService: CIM_ElementCapabilities	45
96	7.3.9	IndicationServiceInitialSettings: CIM_IndicationServiceSettingData.....	45
97	7.3.10	InitialSettingsOfIndicationService: CIM_ElementSettingData	46
98	7.3.11	IndicationFilter: CIM_IndicationFilter	47
99	7.3.12	StaticIndicationFilter: CIM_IndicationFilter	50
100	7.3.13	DynamicIndicationFilter: CIM_IndicationFilter	51
101	7.3.14	IndicationServiceOfIndicationFilter: CIM_ServiceAffectsElement	54
102	7.3.15	IndicationSpecificIndicationFilter: CIM_IndicationFilter	55
103	7.3.16	GlobalIndicationFilter: CIM_IndicationFilter	56
104	7.3.17	StaticFilterCollection: CIM_FilterCollection	57
105	7.3.18	IndicationServiceOfFilterCollection: CIM_OwningCollectionElement.....	60
106	7.3.19	IndicationFilterInFilterCollection: CIM_MemberOfCollection	61
107	7.3.20	FilterCollectionInFilterCollection: CIM_MemberOfCollection	61
108	7.3.21	ProfileSpecificFilterCollection: CIM_FilterCollection.....	62
109	7.3.22	GlobalFilterCollection: CIM_FilterCollection	64
110	7.3.23	ListenerDestination: CIM_ListenerDestination	66
111	7.3.24	IndicationServiceOfListenerDestination: CIM_ServiceAffectsElement.....	70
112	7.3.25	AbstractSubscription: CIM_AbstractIndicationSubscription.....	71
113	7.3.26	FilterSubscription: CIM_IndicationSubscription	75
114	7.3.27	CollectionSubscription: CIM_FilterCollectionSubscription.....	76
115	7.3.28	ProfileOfFilterCollection: CIM_ConcreteDependency	78
116	7.3.29	BasicIndication: CIM_Indication.....	78
117	7.3.30	ReliableIndication: CIM_Indication	81
118	7.3.31	AlertIndication: CIM_AlertIndication.....	82
119	7.3.32	LifecycleIndication: CIM_InstIndication.....	86
120	7.3.33	ListenerDestinationRemovalIndication: CIM_InstDeletion	88
121	7.3.34	SubscriptionRemovalIndication: CIM_InstDeletion.....	88
122	7.4	Reliable indication delivery	89
123	7.4.1	General	89
124	7.4.2	Sequence identifier and sequence identifier lifetime	89
125	7.4.3	WBEM server requirements.....	90
126	7.4.4	WBEM listener requirements	91
127	8	Use Cases.....	93
128	8.1	Object Diagrams	93
129	8.2	LocateIndicationService: Locate the indication service provided by an implementation of this profile.....	97
130	8.3	LocateProfileInIndicationService: Locate the indication service responsible for delivering indications defined by a referencing profile	98
131	8.4	DetermineIndicationServiceCapabilities: Determine the capabilities of an indication service.....	98
132	8.5	ModifyIndicationService: Modify functional aspects of an indication service	99
133	8.6	ListListenerDestinations: List all listener destinations exposed by an implementation	100
134	8.7	SelectListenerDestination: Select an existing listener destination referencing a desired listener	101
135	8.8	CreateListenerDestination: Create a new listener destination	101
136	8.9	FindFreeListenerDestination: Find a free listener destination	102
137			
138			
139			
140			

141 8.10 ModifyListenerDestination: Modify an existing listener destination 103

142 8.11 DeleteListenerDestination: Delete an existing listener destination..... 103

143 8.12 FindIndicationFilter: Find an indication filter covering a particular indication 104

144 8.13 DetermineQueryLanguages: Determine the set of query languages supported for query

145 statements 104

146 8.14 CreateIndicationFilter: Create a dynamic indication filter covering a particular indication 105

147 8.15 ModifyIndicationFilter: Modify a dynamic indication filter..... 106

148 8.16 DeleteIndicationFilter: Delete a dynamic indication filter 106

149 8.17 CheckCollectionCoverage: Check the coverage of a filter collection 107

150 8.18 ObtainNamedCollection: Obtain a named filter collection 108

151 8.19 CreateSubscription: Create a subscription 108

152 8.20 CheckSubscriptions: Determine whether subscriptions exist for a given indication and

153 listener 110

154 8.21 DeleteSubscription: Delete a subscription 110

155 8.22 FindAlertingSystem: Find the system containing a component causing an alert indication... 111

156 8.23 DetermineIndicationGate: Determine the indication gate of an indication..... 111

157 8.24 SubscribeForProfileIndications: Subscribe for all of the indications defined in a

158 referencing profile 113

159 ANNEX A (informative) Profiles defining indications 114

160 ANNEX B (informative) Change Log..... 115

161 **Figures**

163 Figure 1 – Indication related functionality within an implementation 22

164 Figure 2 – Indications Profile: DMTF class adaptation diagram 31

165 Figure 3 – Indications Profile: Indication adaptations and adapted indication classes..... 33

166 Figure 4 – DMTF object diagram: Global and profile-specific filter collections..... 93

167 Figure 5 – DMTF object diagram: Filter collections and contained indication filters..... 95

168 Figure 6 – DMTF object diagram: Static listener destinations 96

169 **Tables**

171 Table 1 – Profile references 17

172 Table 2 – Adaptations 17

173 Table 3 – Features 18

174 Table 4 – IndicationService: Element requirements 38

175 Table 5 – ModifyInstance(): Error reporting requirements 39

176 Table 6 – IndicationSystem: Element requirements 41

177 Table 7 – HostedIndicationService: Element requirements..... 41

178 Table 8 – IndicationsProfileRegistration: Element requirements 42

179 Table 9 – ElementConformsToProfile: Element requirements 43

180 Table 10 – IndicationServiceCapabilities: Element requirements..... 44

181 Table 11 – CapabilitiesOfIndicationService: Element requirements..... 45

182 Table 12 – IndicationServiceInitialSettings: Element requirements..... 46

183 Table 13 – InitialSettingsOfIndicationService: Element requirements..... 47

184 Table 14 – IndicationFilter: Element requirements 48

185 Table 15 – StaticIndicationFilter: Element requirements 50

186 Table 16 – DynamicIndicationFilter: Element requirements 51

187 Table 17 – CreateInstance(): Error reporting requirements..... 51

188 Table 18 – DeleteInstance(): Error reporting requirements 53

189 Table 19 – ModifyInstance(): Error reporting requirements 53

190 Table 20 – IndicationServiceOfIndicationFilter: Element requirements 54

191 Table 21 – IndicationSpecificIndicationFilter: Element requirements 55

192 Table 22 – GlobalIndicationFilter: Element requirements 57

193 Table 23 – GlobalIndicationFilter: Instance requirements for instances covering all alert indications 57

194 Table 24 – GlobalIndicationFilter: Instance requirements for instances covering all lifecycle indications . 57

195 Table 25 – StaticFilterCollection: Element requirements 59

196 Table 26 – IndicationServiceOfFilterCollection: Element requirements 60

197 Table 27 – IndicationFilterInFilterCollection: Element requirements 61

198 Table 28 – FilterCollectionInFilterCollection: Element requirements 62

199 Table 29 – ProfileSpecificFilterCollection: Element requirements 62

200 Table 30 – GlobalFilterCollection: Element requirements 64

201 Table 31 – ListenerDestination Element requirements 67

202 Table 32 – CreateInstance(): Error reporting requirements 68

203 Table 33 – ListenerDestination.DeleteInstance(): Error reporting requirements 69

204 Table 34 – ModifyInstance(): Error reporting requirements 69

205 Table 35 – IndicationServiceOfListenerDestination: Element requirements 70

206 Table 36 – AbstractSubscription: Element requirements 71

207 Table 37 – RepeatNotificationPolicy: Value constraints 72

208 Table 38 – ModifyInstance(): Error reporting requirements 74

209 Table 39 – FilterSubscription: Element requirements 75

210 Table 40 – CreateInstance(): Error reporting requirements 76

211 Table 41 – CollectionSubscription: Element requirements 77

212 Table 42 – CreateInstance(): Error reporting requirements 77

213 Table 43 – ProfileOfFilterCollection: Element requirements 78

214 Table 44 – BasicIndication: Element requirements 80

215 Table 45 – ReliableIndication: Element requirements 81

216 Table 46 – AlertIndication: Element requirements 84

217 Table 47 – LifecycleIndication: Element requirements 87

218 Table 48 – ListenerDestinationRemovalIndication: Element requirements 88

219 Table 49 – SubscriptionRemovalIndication: Element requirements 89

220

221

Foreword

222 The *Indications Profile* (DSP1054) was prepared by the DMTF Architecture Working Group. Version 1.0
223 was prepared by the DMTF WBEM Infrastructure and Protocols Working Group. Versions up to 1.2 were
224 prepared by the WBEM Infrastructure Modeling Working Group.

225 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
226 management and interoperability. For information about the DMTF, see <http://www.dmtf.org>.

227 Acknowledgments

228 DMTF acknowledges the following individuals for their contributions to this document:

229 DMTF acknowledges the following individuals for their contributions to this document:

230 Editor:

- 231 • Michael Johanssen, IBM

232 Contributors:

- 233 • Jim Davis, WBEM Solutions (former editor)
- 234 • Steve Hand, Dell (former editor)
- 235 • Jon Hass, Dell (former editor)
- 236 • David Judkovics, IBM (former editor)
- 237 • Andreas Maier, IBM (former editor)
- 238 • Aaron Merkin, Dell (former editor)
- 239 • Venkat Puvvada, IBM
- 240 • Karl Schopmeyer, DMTF Fellow
- 241 • Hemal Shah, Broadcom (former editor)

242

Introduction

243 The information in this specification should be sufficient for a provider or consumer of this data to
244 unambiguously identify the classes, properties, methods, and values that shall be instantiated to
245 subscribe, advertise, produce, or consume an indication using the DMTF Common Information Model
246 (CIM) Schema.

247 The target audience for this specification is implementers who are writing CIM-based providers or
248 consumers of management interfaces that represent the components described in this document.

249 Document conventions

250 Typographical conventions

251 Any text in this document is in normal text font, with the following exceptions:

- 252 • Document titles are marked in *italics*.
- 253 • Important terms that are used for the first time are marked in *italics*.
- 254 • Terms within the text contain a link to the term definition defined in the "Terms and definitions"
255 clause, enabling easy navigation to the term definition.
- 256 • ABNF rules are in `monospaced font`.

257 ABNF usage conventions

258 Format definitions in this document are specified using ABNF (see [RFC5234](#)), with the following
259 deviations:

- 260 • Literal strings are to be interpreted as case-sensitive Unicode characters, as opposed to the
261 definition in [RFC5234](#) that interprets literal strings as case-insensitive US-ASCII characters.

262 Deprecated material

263 Deprecated material is not recommended for use in new development efforts. Existing and new
264 implementations may use this material, but they shall move to the newer approach as soon as possible.
265 An implementation of this profile in a CIM server shall use any deprecated material as if it were not
266 deprecated, in order to achieve backwards compatibility for clients. Although implementations of clients
267 may use deprecated material, it is recommended that they use the newer approach instead.

268 The following typographical convention indicates deprecated material:

269 DEPRECATED

270 Deprecated material appears here.

271 DEPRECATED

272 In places where this typographical convention cannot be used (for example tables or figures), the
273 "DEPRECATED" label is used alone.

274 Experimental material

275 Experimental material has yet to receive sufficient review to satisfy the adoption requirements set forth by
276 the DMTF. Experimental material is included in this document as an aid to implementers who are
277 interested in likely future developments. Experimental material may change as implementation

278 experience is gained. It is likely that experimental material will be included in an upcoming revision of the
279 specification. Until that time, experimental material is purely informational.

280 The following typographical convention indicates experimental material:

281 **EXPERIMENTAL**

282 Experimental material appears here.

283 **EXPERIMENTAL**

284 In places where this typographical convention cannot be used (for example tables or figures), the
285 "EXPERIMENTAL" label is used alone.

286

Indications Profile

287 1 Scope

288 The *Indications Profile* defines the CIM elements that are used to subscribe for indications of unsolicited
289 events, to advertise the possible indications, and to represent indications used to report events in a
290 managed system.

291 2 Normative references

292 The following referenced documents are indispensable for the application of this document. For dated or
293 versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies.
294 For undated and unversioned references, the latest published edition of the referenced document
295 (including any corrigenda or DMTF update versions) applies.

296 DMTF DSP0004, *CIM Infrastructure Specification 2.6*,
297 http://www.dmtf.org/standards/published_documents/DSP0004_2.6.pdf

298 DMTF DSP0202, *CIM Query Language Specification 1.0*,
299 http://www.dmtf.org/standards/published_documents/DSP0202_1.0.pdf

300 DMTF DSP0207, *WBEM URI Mapping Specification 1.0*,
301 http://www.dmtf.org/standards/published_documents/DSP0207_1.0.pdf

302 DMTF DSP0223, *Generic Operations 1.0*,
303 http://www.dmtf.org/standards/published_documents/DSP0223_1.0.pdf

304 DMTF DSP0228, *Message Registry XML Schema 1.1*,
305 http://schemas.dmtf.org/wbem/messageregistry/1/dsp0228_1.1.xsd

306 DMTF DSP1001, *Management Profile Specification Usage Guide 1.1*,
307 http://www.dmtf.org/standards/published_documents/DSP1001_1.1.pdf

308 DMTF DSP1033, *Profile Registration Profile 1.0*,
309 http://www.dmtf.org/standards/published_documents/DSP1033_1.0.pdf

310 IETF RFC3986, *Uniform Resource Identifier (URI): Generic Syntax, January 2005*,
311 <http://tools.ietf.org/html/rfc3986>

312 IETF RFC5234, *Augmented BNF for Syntax Specifications: ABNF, January 2008*,
313 <http://tools.ietf.org/html/rfc5234>

314 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
315 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>

316 **3 Terms and definitions**

317 In this document, some terms and verbal phrases have a specific meaning beyond the normal English
318 meaning. Those terms and verbal phrases are defined in this clause.

319 The verbal phrases "shall" ("required"), "shall not", "should" ("recommended"), "should not" ("not
320 recommended"), "may", "need not" ("not required"), "can" and "cannot" in this document are to be
321 interpreted as described in [ISO/IEC Directives, Part 2](#), Annex H . The verbal phrases in parenthesis are
322 alternatives for the preceding verbal phrase, for use in exceptional cases when the preceding verbal
323 phrase cannot be used for linguistic reasons. Note that [ISO/IEC Directives, Part 2](#), Annex H specifies
324 additional alternatives. Occurrences of such additional alternatives shall be interpreted in their normal
325 English meaning.

326 The terms "clause", "subclause", "paragraph", "annex" in this document are to be interpreted as described
327 in [ISO/IEC Directives, Part 2](#), clause 5.

328 The terms "normative" and "informative" in this document are to be interpreted as described in [ISO/IEC](#)
329 [Directives, Part 2](#), clause 3. In this document, clauses, subclauses or annexes indicated with
330 "(informative)" do not contain normative content. Notes and examples are always informative elements.

331 The terms defined in [DSP0004](#), [DSP0223](#) and [DSP1001](#) apply to this document. The following additional
332 terms are used in this document.

333 **3.1**

334 **alert indication**

335 an indication that indicates an event related to the managed environment
336 For details, see 6.1.2.2.

337 **3.2**

338 **client**

339 a WBEM client that exploits applicable portions of this profile
340 For details, see [DSP1001](#).

341 **3.3**

342 **coverage**

343 the set of indications that can pass an indication gate
344 For details, see 6.2.2 and 6.3.2.

345 **3.4**

346 **defined coverage**

347 the coverage specified by a profile for static filter collections through normative statements
348 For details, see 6.3.3.

349 **3.5**

350 **dynamic indication filter**

351 an indication filter whose lifecycle is controlled by a client

352 **3.6**

353 **event**

354 an observable occurrence of a phenomenon of interest
355 For details, see 6.1.

- 356 **3.7**
357 **filter collection**
358 an indication gate that may contain other indication gates such as indication filters or other filter
359 collections
360 For details, see 6.3.
- 361 **3.8**
362 **global indication filter**
363 an indication filter that covers large sets of indications, such as all alert indications
364 For details, see 6.2.5.
- 365 **3.9**
366 **global filter collection**
367 a filter collection that covers large sets of indications, such as all lifecycle indications
368 For details, see 6.3.3.5.
- 369 **3.10**
370 **implementation**
371 a WBEM server that implements applicable portions of this profile and of referencing profiles
372 For details, see [DSP1001](#).
- 373 **3.11**
374 **indication**
375 the notification about an event that occurred
376 For details, see 6.1.
- 377 **3.12**
378 **indication delivery**
379 the process of delivering indications from an implementation to a listener
380
- 381 **indication filter**
382 an indication gate whose coverage is defined through a query statement
383 For details, see 6.2
- 384 **3.13**
385 **indication filtering**
386 the process of selecting indications based on filtering rules applied by indication gates, such that only
387 indications within the coverage of the indication gate pass the indication gate
- 388 **3.14**
389 **indication gate**
390 a managed element that filters indications such that only indications within its coverage pass. Indication
391 gates can serve as targets for subscriptions, and control which indications are delivered to subscribed
392 listeners.
- 393 **3.15**
394 **indication generation**
395 the process of creating an indication as the event that the indication is designed to report occurs
- 396 **3.16**
397 **indication origin**
398 the namespace out of that the indication originates
399 For details, see 6.1.2.4.

- 400 **3.17**
401 **indication service**
402 a component within a WBEM server for indication related processing, including handling of subscriptions
403 and delivery of indications to a WBEM listener
- 404 **3.18**
405 **indication system**
406 a system that hosts a WBEM server with one or more indication services
407 For details, see 6.6.
- 408 **3.19**
409 **indication-specific indication filter**
410 a static indication filter that covers a particular indication specified in a profile
411 For details, see 6.2.4.
- 412 **3.20**
413 **Interop namespace**
414 a namespace containing CIM instances representing specific capabilities of a WBEM server
415 Examples include CIM_RegisteredProfile instances representing specific versions of profiles or
416 CIM_IndicationFilter instances representing indication filters. For details, see [DSP1033](#).
- 417 **3.21**
418 **lifecycle indication**
419 an indication indicating an event related to the lifecycle of CIM instances or CIM classes; for details,
420 see 6.1.2.3.
- 421 **3.22**
422 **listener**
423 a WBEM listener that implements applicable portions of this profile
424 For details, see [DSP1001](#).
- 425 **3.23**
426 **listener destination**
427 an entity that maintains a reference to a listener within an implementation; for details, see 6.4.5..
- 428 **3.24**
429 **profile-specific filter collection**
430 a static filter collection that covers all indications of a particular type defined in a profile
431 For details, see 6.3.3.4.
- 432 **3.25**
433 **query statement**
434 a statement expressed in a query language used to describe either (a part of) an event or the coverage of
435 an indication filter
- 436 **3.26**
437 **referencing profile**
438 a profile referencing this profile
439 Note that [DSP1001](#) requires each profile that defines indications to reference this profile.
- 440 **3.27**
441 **reliable indication**
442 an indication containing a sequence identifier enabling listeners to detect duplicate, missing, or out-of-
443 order indications

- 444 For details, see 6.1.5 and 7.4.
- 445 **3.28**
- 446 **repeated indication**
- 447 an indication that reports the same event as a previous indication
- 448 For details, see 6.1.6.
- 449 **3.29**
- 450 **repeated indication delivery**
- 451 the delivery of repeated indications
- 452 Repeated indication delivery typically occurs if the reported event describes a persistent situation such as
- 453 exceeding a threshold value.
- 454 **3.30**
- 455 **sequence identifier**
- 456 data element with a reliable indication that ensures unique identification of the reliable indication
- 457 A sequence identifier is composed of a sequence context and a sequence number
- 458 For details, see 7.4.2.
- 459 **3.31**
- 460 **sequence identifier lifetime**
- 461 a maximum time interval maintained by an implementation implementing reliable indications within which
- 462 the implementation retries failed indication delivery attempts
- 463 For details, see 7.4.2.
- 464 **3.32**
- 465 **static filter collection**
- 466 a filter collection whose lifecycle is controlled by the implementation, that is uniquely identifiable and for
- 467 which a defined coverage is established
- 468 For details, see 6.3.3.
- 469 **3.33**
- 470 **static indication filter**
- 471 an indication filter whose lifecycle is controlled by the implementation
- 472 **3.34**
- 473 **subscription**
- 474 the mechanism whereby a client registers a listener for the delivery of indications from an implementation
- 475 **3.35**
- 476 **this profile**
- 477 a short term for the Indications profile, the profile specified in this specification document (DSP1054)
- 478 **3.36**
- 479 **WBEM client**
- 480 a CIM client (see [DSP0004](#)) that supports a WBEM protocol
- 481 For details, see [DSP1001](#).
- 482 **3.37**
- 483 **WBEM listener**
- 484 a CIM listener (see [DSP0004](#)) that supports a WBEM protocol
- 485 For details, see [DSP1001](#).

486 **3.38**
487 **WBEM server**
488 a CIM server (see [DSP0004](#)) that supports a WBEM protocol
489 For details, see [DSP1001](#).

490 **4 Symbols and abbreviated terms**

491 **4.1**
492 **CQL**
493 CIM Query Language

494 **4.2**
495 **QoS**
496 Quality of service

497 **4.3**
498 **URI**
499 Uniform Resource Identifier

500 **4.4**
501 **WBEM**
502 Web Based Enterprise Management

503 **5 Synopsis**

504 **Profile name:** Indications

505 **Version:** 1.2.0

506 **Organization:** DMTF

507 **Profile type:** Component

508 **Schema version:** 2.25

509 **Central class adaptation:** IndicationService (see 7.3.2)

510 **Scoping class adaptation:** IndicationSystem (see 7.3.3)

511 **Scoping algorithm:** HostedIndicationService (see 7.3.4)

512 This profile extends the management capabilities defined in referencing profiles by adding the capability
513 to subscribe for indications of unsolicited events, and to notify about such events by means of sending
514 indications from the implementation to a listener. This profile defines the required content of indications
515 defined in referencing profiles.

516 Table 1 lists the profile references defined by this profile.

517 **Table 1 – Profile references**

Profile reference name	Profile name	Organization	Version	Relationship	Description
ProfileRegistration	Profile Registration	DMTF	1.0	Mandatory	Registration of this profile; the central class profile advertisement methodology is mandated by this profile; for details, see 7.3.6.

518 Table 2 lists the class adaptations that are defined in this profile.

519 **Table 2 – Adaptations**

Adaptation	Elements	Requirement	Description
Instantiated and embedded class adaptations			
IndicationService	CIM_IndicationService	Mandatory	See 7.3.2.
IndicationSystem	CIM_System	Mandatory	See 7.3.3.
HostedIndicationService	CIM_HostedService	Mandatory	See 7.3.4.
IndicationsProfileRegistration	CIM_RegisteredProfile	Mandatory	See 7.3.5.
ElementConformsToProfile	CIM_ElementConformsToProfile	Mandatory	See 7.3.6.
IndicationServiceCapabilities	CIM_IndicationServiceCapabilities	Conditional	See 7.3.7.
CapabilitiesOfIndicationService	CIM_ElementCapabilities	Conditional	See 7.3.8.
IndicationServiceInitialSettings	CIM_IndicationServiceSettingData	Conditional	See 7.3.9.
InitialSettingsOfIndicationService	CIM_ElementSettingData	Conditional	See 7.3.10.
IndicationFilter	CIM_IndicationFilter	See derived adaptations	See 7.3.11.
StaticIndicationFilter	CIM_IndicationFilter	See derived adaptations	See 7.3.12.
DynamicIndicationFilter	CIM_IndicationFilter	Conditional	See 7.3.13.
IndicationServiceOfIndicationFilter	CIM_ServiceAffectsElement	Mandatory	See 7.3.14.
IndicationSpecificIndicationFilter	CIM_IndicationFilter	Optional	See 7.3.15.
GlobalIndicationFilter	CIM_IndicationFilter	Mandatory	See 7.3.16.
StaticFilterCollection	CIM_FilterCollection	See derived adaptations	See 7.3.17.
IndicationServiceOfFilterCollection	CIM_OwningCollectionElement	Mandatory	See 7.3.18.
IndicationFilterInFilterCollection	CIM_MemberOfCollection	Conditional	See 7.3.19.
FilterCollectionInFilterCollection	CIM_MemberOfCollection	Conditional	See 7.3.20.
ProfileSpecificFilterCollection	CIM_FilterCollection	Optional	See 7.3.21.
GlobalFilterCollection	CIM_FilterCollection	Mandatory	See 7.3.22.
ListenerDestination	CIM_ListenerDestination	Mandatory	See 7.3.23.
IndicationServiceOfListener-Destination	CIM_ServiceAffectsElement	Mandatory	See 7.3.24.
AbstractSubscription	CIM_AbstractIndication-Subscription	See derived adaptations	See 7.3.25.
FilterSubscription	CIM_IndicationSubscription	Conditional	See 7.3.26.

Adaptation	Elements	Requirement	Description
CollectionSubscription	CIM_FilterCollectionSubscription	Mandatory	See 7.3.27.
ProfileOfFilterCollection { D }	CIM_ConcreteDependency	Mandatory	See 7.3.28.
Indications and exceptions			
BasicIndication	CIM_Indication	See derived adaptations	See 7.3.29.
ReliableIndication	CIM_Indication	See derived adaptations	See 7.3.30.
AlertIndication	CIM_AlertIndication	See derived adaptations	See 7.3.31.
LifecycleIndication	CIM_InstIndication	See derived adaptations	See 7.3.32.
ListenerDestination-RemovalIndication	CIM_InstDeletion	Optional	See 7.3.33.
SubscriptionRemovalIndication	CIM_InstDeletion	Optional	See 7.3.34.

520 Table 3 lists the features that are defined in this profile.

521 **Table 3 – Features**

Feature name	Granularity	Requirement	Description
DynamicIndicationFilters	IndicationService instance	Optional	See 7.2.1.
IndicationServiceInitialSettingsExposed	IndicationService instance	Optional	See 7.2.2.
IndicationServiceModification	IndicationService instance	Optional	See 7.2.3.
ReliableIndications	IndicationService instance	Optional	See 7.2.4.
SuppressRepeatNotificationPolicy	Profile implementation	Optional	See 7.2.5.
DelayRepeatNotificationPolicy	Profile implementation	Optional	See 7.2.6.
IndividualFilterSubscription	IndicationFilter instance	Optional	See 7.2.7.
FilterCollectionCoverageExposure	StaticFilterCollection instance	Conditional	See 7.2.8.

522 **6 Description**

523 This profile defines the concept of indications as a means to notify listeners about events occurring in the
524 managed environments addressed by referencing profiles. This profile establishes basic reusable
525 elements enabling referencing profiles to specify indications that report events occurring in their managed
526 environments. For example, this profile defines reusable adaptations of CIM classes by defining
527 requirements or constraints on suitable properties and methods, by defining required relationships, and
528 by defining the modeled object types in the managed environment.

529 Furthermore, this profile defines how clients can subscribe listeners for the delivery of indications, and
530 how clients can monitor and control certain aspects of the behavior of implementations of this profile,
531 such as the number of retry attempts or the retry delay when the implementation is unable to deliver
532 indications.

533 This profile also defines mechanisms for the reliable delivery of indications.

534 **6.1 Events and indications**

535 **6.1.1 Events**

536 An event is the observable occurrence of a phenomenon of interest.

537 Events could be distinguished into root events and secondary events.

538 Root events are events directly related the managed environment; they may be related to a managed
539 object.

540 Secondary events are events that are effected by or occur as a consequence of root events. For
541 example, a root event could be the emergence of a fire on a house. Smoke or heat are both possible
542 effects or, in other words, secondary events, caused by the fire.

543 Furthermore, if a managed object is represented in CIM, the model changes resulting from the change of
544 a managed object may be visible through corresponding changes in its CIM representation.

545 **6.1.2 Indications**

546 **6.1.2.1 General**

547 An indication is a notification about an event. It is possible that an indication only reports an aspect of the
548 event and not the entire event. Therefore, multiple indications may be reported in context of a particular
549 event.

550 For example, an indication could directly report the root event that a house has caught fire. In addition, or
551 alternatively, respective indications could separately report secondary events (or effects) caused by the
552 fire, such as that smoke or heat are observed.

553 Accordingly, if a managed object is represented in CIM, an indication could directly report the root event
554 related to the managed object. In addition, or alternatively, respective indications could separately report
555 events (or effects) caused by the root event, such that a CIM instance representing an aspect of the
556 managed object was created, modified or deleted.

557 Reporting events from the managed environment is typically facilitated by means of alert indications,
558 whereas reporting events from the CIM model is typically facilitated by means of lifecycle indications.

559 **6.1.2.2 Alert indications**

560 Alert indications are indications that provide notification about root events (see 6.1.1). If a reported event
561 relates to a managed object, that managed object may or may not have a representation in CIM. Some

562 types of alert indications can also contain information about or refer to corresponding changes in the CIM
563 representation where that is available.

564 6.1.2.3 Lifecycle indications

565 Lifecycle indications are indications that provide notification about events (see 6.1.1) related to the
566 lifecycle of CIM instances and CIM classes, such as their creation, deletion or modification.

567 Only lifecycle events related to the creation, deletion, or modification of CIM instances are within the
568 scope of this profile.

569 NOTE The CIM schema defines the CIM_InstIndication class as the base class for indications reporting lifecycle
570 events and other model-related events, such as the execution of methods or the execution of read
571 operations; reporting the latter kinds of events is not addressed in this profile.

572 Lifecycle events related to CIM instances are reported using instances of adaptations of the
573 CIM_InstCreation, CIM_InstDeletion, or CIM_InstModification classes.

574 It is important to realize that lifecycle events are events (see 6.1.1) in the CIM model, reflecting
575 corresponding events in the managed environment. This applies regardless of whether or not a change
576 was requested by means of a CIM operation; CIM instances are required to always correctly represent
577 (an aspect of) the actual state of a managed object, and thus can only change if the represented (aspect
578 of the) managed object changed.

579 [DSP1001](#) defines the existence of CIM instances as a logical concept that ties the existence of CIM
580 instances to the existence of the represented managed object in the managed environment (instead of
581 tying the existence of CIM instances to a physical representation such as a repository entry). By that
582 definition the creation of a CIM instance logically occurs when the represented managed object is added
583 to the managed environment, and the deletion of a CIM instance logically occurs when the represented
584 managed object is removed from the managed environment.

585 With that definition, a CIM instance logically exists even if the WBEM server containing its implementation
586 is inactive, or does temporarily not have access to the managed environment containing the represented
587 managed object. If a WBEM server is inactive when a managed object is added to the managed
588 environment, the CIM instance(s) representing (an aspect of) that managed object still are assumed to be
589 "logically" created exactly at that point in time; however, because the WBEM server is inactive, no
590 lifecycle indications are sent. Furthermore, when the WBEM server is started later on, sending lifecycle
591 indications about lifecycle events occurring while the WBEM server was inactive is not to be made up for.
592 Similarly, when a WBEM server is initially started, lifecycle indications about instances initially existing
593 within that WBEM server are not to be sent. So the [DSP1001](#) based definition of instance existence
594 provides for not having to indicate the creation / deletion of CIM instances every time a WBEM server is
595 activated or deactivated, and avoids requiring a WBEM server to determine which CIM instances were
596 created / deleted / modified while it was inactive.

597 With the [DSP1001](#) based definition of instance existence, clients may exploit lifecycle indications as a
598 means to monitor the existence of the represented managed object in the managed environment.
599 However, clients cannot rely on indications as the sole means to track the lifecycle of managed objects in
600 the managed environment. At least initially, and after every WBEM server restart, clients actively need to
601 inspect (by means of invoking respective operations) the CIM model of the managed environment for
602 changes that occurred while the WBEM server was inactive. If reliable indications (see 6.1.5) are
603 implemented, a change of the value of the SequenceContext property in the stream of indications arriving
604 at a particular listener from a particular WBEM server may be used as an indicator that a WBEM server
605 restart occurred; for details, see 7.3.30.2.2, and the CIM schema definition of the CIM_Indication class.

606 A CIM model can represent different aspects of a particular managed object through several instances of
607 different CIM classes. Consequently, one event in the managed environment can be related to multiple
608 events in the CIM model of the managed environment, such as changes in several CIM instances, each
609 of which could be reported through a separate lifecycle indication.

610 As an example, consider a managed environment composed of systems and their components. If a
611 component such as a fan is added to one of these systems, this would be constitute an event in the
612 managed environment and could be reported by means of an alert indication. Alternatively, or in addition,
613 if the added fan is represented by a CIM_Fan instance, the creation of that CIM_Fan instance could be
614 reported by means of a lifecycle indication.

615 **6.1.2.4 Origin of indications**

616 The origin of an indication is defined as the local namespace in context of that the indication is generated;
617 for details, see 7.3.29.3.

618 The CIM representation of an indication as defined by the CIM_Indication class does not reflect the origin
619 namespace. Nevertheless, the process of indication filtering (see 6.1.4) is required to consider the origin
620 namespace of an indication; for details, see 7.3.11.2.

621 **6.1.3 Definition of events and indications in referencing profiles**

622 Referencing profiles may define events separately through normative text, or as part of the definition of
623 indication adaptations reporting the event.

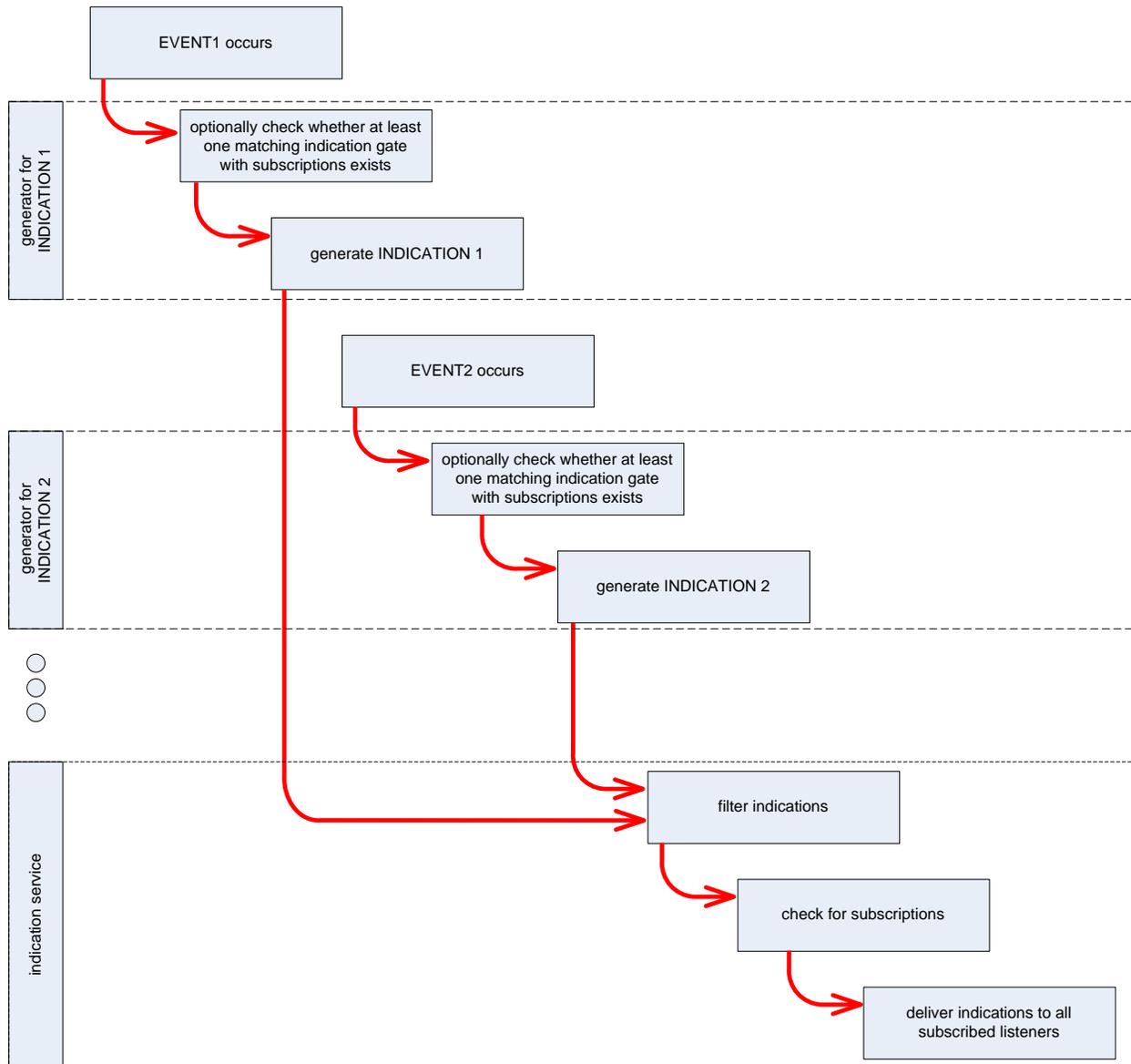
624 NOTE Defining events separately is particularly useful if multiple indications reporting the same event are
625 defined. However, if an event is only reported through one indication, the event definition as part of the
626 definition of the indication adaptation is more compact.

627 This profile defines several basic indication adaptations for the use by referencing profiles that define
628 indications:

- 629 • The BasicIndication adaptation requires the reported event to be specified by means of a query
630 statement; for details, see 7.3.29.2.
- 631 • The AlertIndication adaptation refines the BasicIndication adaptation for alert indications. It
632 refines the definition of the query statement, delegating the event definition to an alert message
633 defined in a message registry. For details, see 7.3.31.
- 634 • The LifecycleIndication adaptation refines the BasicIndication adaptation for lifecycle
635 indications. A lifecycle indication refers to the CIM instance for which it reports a lifecycle event.
636 The profile defining the lifecycle indications defines for which class adaptations respective
637 lifecycle indications are reported. For details, see 7.3.32.

638 **6.1.4 Indication generation, indication filtering, and indication delivery**

639 The indication related functionality within an implementation can be structured into indication generation,
640 indication filtering and indication delivery. This is detailed in Figure 1.



641

642

Figure 1 – Indication related functionality within an implementation

643 Indication generation is the process of creating an indication as the event that the indication is designed
 644 to report occurs. As shown in Figure 1, this functionality is typically implemented separately for each
 645 indication, because it depends on the distinct event reported through each particular indication.

646 Optionally, in order to avoid the generation of indications for which no listeners are subscribed, part of
 647 indication filtering can already occur at indication generation time, such that an indication is only
 648 generated if at least one indication gate exists that has a coverage covering the indication to be
 649 generated, and that has subscribed listeners; for details, see 7.3.29.5. However, even in this case
 650 (complete) indication filtering is still required in order to ensure that the generated indication is checked
 651 against every existing indication gate.

652 After an indication is generated it is subjected to indication filtering. Indication filtering is the process of
 653 selecting indications based on specific filtering rules applied by indication gates, such that only indications
 654 within the coverage of the indication gate pass. This functionality is typically implemented in common

655 independent of the implementation of individual indications; however, it depends on indication gates that
656 may be provided by implementations of referencing profiles. For details, see 7.3.11.2 and 7.3.17.2.

657 Indication delivery is the process of delivering filtered indications from an implementation to a listener.
658 This profile defines rules for the delivery of indications as part of adaptations modeling indications
659 themselves, as part of adaptations modeling indication gates such as indication filters or filter collections,
660 and as part of adaptations modeling subscriptions and listener destinations. For details, see 7.3.23.2 and
661 7.3.25.2.

662 **6.1.5 Reliable indication delivery**

663 Reliable indication delivery is an optional extension of indication delivery that aims to

- 664 • enable implementations to discover and retry unsuccessful indication deliveries, and
- 665 • enable listeners to detect duplicate, missing, or out-of-order indications, and to re-order
666 indications that arrive out of order. This includes the discovery of server restarts.

667 The ReliableIndication adaptation (see 7.3.30) models reliable indications, and additional requirements
668 are specified in 7.4.

669 **6.1.6 Avoidance of repeated indication delivery**

670 **6.1.6.1 General**

671 This profile defines policies for the avoidance of repeated indication delivery (see 3.29). Policies for
672 avoiding repeated indication delivery aim at preventing the implementation from flooding subscribed
673 listeners with large amounts of repeated indications. This is a typical scenario if an event models a
674 persistent situation, such as exceeding a threshold value.

675 For example, consider an indication modeled to report disk i/o errors. If a disk generates i/o errors at a
676 high rate, the implementation would be required to generate a respective amount of indications and
677 deliver them to subscribed listeners.

678 In order to avoid flooding subscribed listeners with such redundant indications, three policies are modeled
679 in this profile, as detailed in 6.1.6.2, 6.1.6.3 and 6.1.6.4.

680 The effective policy for the suppression of repeated indication delivery is determined at the level of
681 subscriptions (see 6.4.1). For a particular subscription, the determination whether an indication passing
682 the indication gate referenced by that subscription is a repeated indication — that is, an indication
683 reporting the same event — of a first indication is made as follows: The first indication starts a monitoring
684 time interval. Any indication passing the referenced indication gate during that monitoring time interval is
685 considered a repeated indication if it is equal with the first indication except for the identification and the
686 generation time.

687 **NOTE** The identification of indications as modeled by the BasicIndication adaptation (see 7.3.29) is exposed by
688 the value of the IndicationIdentifier property, and the generation time is exposed by the value of the
689 IndicationTime property.
690 Version 1.1 of this profile also considered the values of the SequenceContext and the SequenceNumber
691 properties (see 7.3.30.2.2 and 7.3.30.2.3) for the determination of repeated indications. However, the
692 values of these properties are specific for listener destinations. Once these values were determined for a
693 particular indication, that indication must be sent to the referenced listener in order to ensure a continuous
694 and homogeneous stream of indications, thereby enabling reliable indication delivery. Thus, the
695 suppression of repeated indication delivery needs to occur before reliable indication processing, and the
696 determination of repeated indications needs to occur without considering these values.

6.1.6.2 No repeated indication delivery avoidance policy

697 With this policy in effect, no measures against repeated indication delivery are taken (see the CIM
698 schema description of the value 2 (None) for the RepeatNotificationPolicy property of the
699 CIM_AbstractIndicationSubscription class).
700

6.1.6.3 Suppress repeated indication delivery avoidance policy

701 This policy is modeled by means of the SuppressRepeatNotificationPolicy feature (see 7.2.5, and the CIM
702 schema description of the value 3 (Suppress) for the RepeatNotificationPolicy property of the
703 CIM_AbstractIndicationSubscription class).
704

705 With this policy in effect, the implementation with the delivery of a first indication starts a monitoring time
706 interval. If during that monitoring time interval repeated indications of the first indication accrue, these are
707 likewise delivered up to a predefined threshold. If the threshold is reached while the monitoring time
708 interval is not expired, the delivery of further repeated indications is suppressed until the monitoring time
709 interval expires. After the time interval has expired, the cycle is repeated with the next accruing repeated
710 indication.

6.1.6.4 Delayed indication delivery avoidance policy

711 This policy is modeled by the DelayRepeatNotificationPolicy feature (see 7.2.6, and the CIM schema
712 description of the value 4 (Delay) for the RepeatNotificationPolicy property of the
713 CIM_AbstractIndicationSubscription class).
714

715 With this policy in effect, the implementation with a first accruing indication starts a specified monitoring
716 time interval; however, the first indication is not delivered at that point in time. Only if during that
717 monitoring time interval a specified number of repeated indications of the first indication accrue, the
718 implementation delivers the first indication, but suppresses delivering the remaining accrued indications
719 during the monitoring time interval, and then waits for a separately specified delay time interval. After that,
720 or if the specified number of repeated indications did not accrue during the monitoring time interval, the
721 cycle is repeated, using the next accruing repeated indication as the next first indication.

722 Note that with this policy it is possible that no indications are actually delivered if the specified number of
723 repeated indications does not accrue during the monitoring time interval.

6.2 Indication filters

6.2.1 General

724 Indication filters are a special kind of indication gate. The main purposes of indication filters are as
725 follows:

- 726 • Indication filters can serve as targets for subscriptions; for details on subscriptions, see 6.4.
- 727 • Indication filters filter indications such that only indications within the coverage of the indication
728 filter pass for further processing; for details on defining and exposing the indication filter
729 coverage, see 6.2.2.
- 730 • Dynamic indication filters enable clients to establish indication filters with client specified
731 coverage within the implementation; for details, see 6.2.6.
- 732 • If defined in profiles, indication filters can represent an implementation's ability to generate
733 respective indications. However, in general it is not possible to conclude from the existence of
734 an indication filter that an implementation actually generates and delivers any indications
735 covered by that indication filter.
736
737

738 The lifecycle of indication filters is controlled by the implementation. For static indication filters (see 6.2.3),
739 this applies without restrictions; the concept of dynamic indication filters (see 6.2.6) provides for clients

740 being able to prompt the implementation for the creation, modification or deletion of dynamic indication
741 filters.

742 Generally the existence of an indication filter does not imply that any of the indications covered by the
743 indication filter is actually implemented. However, referencing profiles may define amended semantics for
744 indication filters. For details, see 7.3.11.2.

745 Listeners subscribed to an indication gate must be prepared to process any indication within the coverage
746 of the indication gate.

747 6.2.2 Indication filter coverage

748 The coverage of an indication filter is the set of indications that can pass the indication filter; it is specified
749 through an indication filter query statement and a set of namespaces identifications that identify the
750 namespaces out of which indications are filtered. In other words, only indications that originate (see
751 6.1.2.4) in one of the identified namespaces, and match the query statement pass the indication filter. For
752 details, see 7.3.11.2.

753 A indication filter query statement identifies source classes, selects properties, and specifies logic that is
754 used to combine instances of those classes containing the selected property values as part of generated
755 indications.

756 A indication filter query statement is defined using the rules of a query language, for example the CIM
757 Query Language (CQL) (see [DSP0202](#)). Profiles that define indication filters specify the exact string that
758 defines the indication filter query statement.

759 Clients capable of inspecting query statements thereby can learn about the coverage of respective
760 indication filters.

761 Following are examples of properly formatted CQL indication filter query statements:

762 EXAMPLE 1:

```
763 SELECT * FROM CIM_AlertIndication
```

764 This indication filter query statement covers all alert indications. The selection of all properties
765 exposed by the CIM_AlertIndication class indicates that values of these properties are present
766 in CIM_AlertIndication instances delivered to listeners. However, note that generally the value
767 Null is admissible unless otherwise required.

768 EXAMPLE 2:

```
769 SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA  
770 CIM_StorageVolume
```

771 This indication filter query statement covers lifecycle indications reporting the creation of
772 CIM_StorageVolume instances representing newly created storage volumes within the
773 managed environment. This is because the schema definition of the CIM_InstCreation
774 indication states that it indicates the creation of a new CIM instance (of any class), and the
775 WHERE clause limits that to instances of the CIM_StorageVolume class.

776 The selection of all properties exposed by the CIM_InstCreation class indicates that values of
777 these properties are present in CIM_InstCreation instances delivered to listeners. The schema
778 definition of the CIM_InstCreation indication requires that the value of the SourceInstance
779 property contains a copy of the new instance (the CIM_StorageVolume instance in this case).
780 However, with respect to other property values, again note that generally the value Null is
781 admissible unless otherwise required.

782 **EXAMPLE 3:**

```
783 SELECT * FROM CIM_AlertIndication WHERE OwningEntity = 'DMTF' AND
784 MessageID = 'SVPC0123'
```

785 This indication filter query statement covers one alert indication. The related event is defined by
786 an alert message defined in a message repository. The value of the `OwningEntity` property
787 identifies DMTF as the organization owning the message registry. The value of the `MessageID`
788 property allows identifying the alert message within the owning organization; for details, see
789 7.3.31.

790 **EXAMPLE 4:**

```
791 SELECT * FROM CIM_AlertIndication WHERE OwningEntity = 'DMTF' AND
792 MessageID LIKE 'SVPC0123|SVPC0124|SVPC0125'
```

793 This indication filter query statement covers a closed set of alert indications. Note that the use of
794 the `LIKE` expression implies "full like extended regular expressions" as defined in [DSP0202](#).

795 **6.2.3 Static indication filters**

796 Static indication filters are provided by an implementation, that is, their lifecycle and coverage is
797 controlled solely by the implementation, and clients are not able to create or delete static indication filters.

798 Profiles define the requirements for the CIM representation of static indication filters along with a
799 requirement level, such as mandatory, conditional, or optional. In addition, WBEM servers may expose
800 `CIM_IndicationFilter` instances representing static indication filters that are not defined by a profile.

801 Profiles define the coverage of static indication filters (that is, the set of covered indications) through a
802 query statement (see 6.2.2). There is a certain degree of flexibility in defining the indication filter coverage
803 by means of a query statement:

- 804 • Indication filters that cover more than one indication

805 A referencing profile might require an indication filter of this kind in the case where one or more
806 indications covered by that indication filter are implemented.

- 807 • Indication filters that cover exactly one indication

808 This is achieved by specifying a "WHERE" clause as part of the indication filter query statement
809 that restricts the selected indication class to one particular indication. A referencing profile might
810 require an indication filter of this kind for the case "if and only if" the covered indication is
811 implemented. Only in this very special case clients that are aware of that profile definition upon
812 detection of the representation of that particular indication filter would know that the covered
813 indication is actually implemented.

814 Static indication filters are uniquely identified by means of a naming convention that involves the name of
815 the organization defining the profile, the name of this profile and a string that is required to be unique
816 within the implementation of this profile; for details, see 7.3.12.

817 Filter collections provide a means for aggregating the coverage of indication filters and other filter
818 collections; see 6.3.

819 **6.2.4 Indication-specific indication filters**

820 Indication-specific filters address the needs of clients requiring notifications about events reported by
821 particular indications specified in a profile. Indication-specific indication filters are a specialization of static
822 indication filters, and are designed to cover one or more of the indications specified in a referencing
823 profile or in this profile. For details, see 7.3.15.

824 One central purpose of indication-specific indication filters is contributing to the defined coverage of
825 profile-specific filter collections; see 6.3.3.

826 **6.2.5 Global indication filters**

827 Global indication filters address the needs of clients requiring notifications about large sets of events,
828 irrespective of a profile context. Global indication filters are a specialization of static indication filters
829 (see 6.2.3), and are designed to cover large sets of indications, such as:

- 830 • All alert indications
- 831 • All lifecycle indications reporting the creation of a CIM instance
- 832 • All lifecycle indications reporting the modification of a CIM instance
- 833 • All lifecycle indications reporting the deletion of a CIM instance

834 For details, see 7.3.16.

835 **6.2.6 Dynamic indication filters**

836 The creation, deletion and modification of dynamic indication filters can be requested by clients and is
837 then performed by the implementation. If suitable static indication filters do not exist within an
838 implementation, clients can request the creation of dynamic indication filters with a coverage that is
839 specifically tailored to the notification requirements of one or more listeners. However, the implementation
840 of dynamic indication filters is expensive. Not all implementations, especially footprint-sensitive
841 implementations, will be able to implement dynamic indication filters. For that reason this profile models
842 dynamic indication filters in the form of the optional DynamicIndicationFilters feature; for details, see 7.2.1

843 Even if dynamic indication filters are implemented, clients should first look for existing indication filters or
844 filter collections that might satisfy listener notification requirements, before attempting to create a dynamic
845 indication filter. Adding unnecessary dynamic indication filters may adversely affect the performance of
846 indication delivery by the implementation.

847 **6.3 Filter collections**

848 **6.3.1 General**

849 Filter collections are a special kind of indication gate designed to contain other indication gates; the
850 contained indication gates may or may not be represented in CIM.

851 This profile only models static filter collections (see 6.3.3). Dynamic filter collections, that is, filter
852 collections that could be created, deleted and modified by clients, are not addressed by this profile.

853 The main purposes of filter collections are:

- 854 • Filter collections can serve as targets for subscriptions; for details on subscriptions, see 6.4.
- 855 • Filter collections filter indications according to their coverage; for details on defining and
856 exposing the coverage of filter collections, see 6.3.2.
- 857 • If defined in profiles, filter collections can represent an implementation's ability to generate
858 respective indications. However, in general it is not possible to conclude from the existence of a
859 filter collection that an implementation actually generates and delivers any indications covered
860 by that filter collection.

861 **6.3.2 Filter collection coverage**

862 The coverage of a filter collection determines the actual filtering rules for that filter collection; it is defined
863 as the aggregated coverage of all contained indication gates. For details, see 7.3.17.2.

864 **6.3.3 Static filter collections**

865 **6.3.3.1 General**

866 Static filter collections are filter collections whose lifecycle is controlled by the implementation, that are
867 uniquely identifiable, and for which a defined coverage can be established.

868 **6.3.3.2 Unique identification**

869 Unique identification of static filter collections is achieved through establishing a naming convention. The
870 naming convention enables clients to identify static filter collections about which they have prior
871 knowledge. For details on specifying the unique identification, see 7.3.17.4.2.

872 **6.3.3.3 Defined coverage**

873 The concept of the defined coverage addresses the need to reduce the memory footprint of embedded
874 implementations. It allows defining the coverage of static filter collections by means of specification in
875 profiles, but without requiring the CIM representation of contained indication gates. The knowledge about
876 the defined coverages of static filter collections specified in profiles can be built into clients, such that the
877 clients know the coverage of those static filter collections in advance, instead of determining the coverage
878 through the inspection of the CIM representation of contained indication gates. For details on specifying
879 the defined coverage of static filter collections, see 7.3.17.3.

880 **6.3.3.4 Profile specific filter collections**

881 Profile-specific filter collection address the needs of clients requiring notifications about events reported
882 by the indications specified in a particular profile. Profile specific filter collections are a specialization of
883 static filter collections. The defined coverage of a profile-specific filter collection covers all indications of a
884 particular type (that is, all alert indications or all lifecycle indications) defined in a profile. For details, see
885 7.3.21.

886 **6.3.3.5 Global filter collections**

887 Global filter collections address the needs of clients requiring notifications about large sets of events.
888 Global filter collections are a specialization of static filter collections.

889 The defined coverage of global filter collections covers large sets of indications, such as

- 890 • All alert indications
- 891 • All alert indications specified in profiles
- 892 • All lifecycle indications
- 893 • All indications specified in profiles
- 894 • All alert indications specified in profiles
- 895 • All lifecycle indications specified in profiles

896 For details, see 7.3.22.

897 **6.4 Subscriptions, listeners, and listener destinations**

898 **6.4.1 Subscriptions**

899 Subscriptions model a mechanism that enables clients to register listeners at an indication gate for the
900 delivery of indications that are within the coverage of that indication gate.

901 Clients need to perform three steps in order to subscribe a listener for the delivery of indications:

- 902 1) Determine if there is an existing indication gate covering the desired indication set. If an
903 appropriate indication gate does not exist, and the support for dynamic indication filters is
904 implemented, the client could create dynamic indication filters (see 6.2.6).
- 905 2) Determine if a listener destination referencing the listener already exists within the
906 implementation. If such a listener destination does not yet exist, and the support for creating or
907 modifying listener destinations is implemented, the client could create a new listener destination
908 or modify an existing listener destination.
- 909 3) Create a subscription that relates the listener destination with the indication gate.
- 910 After it is created, a subscription results in indications being delivered to the listener that is referenced by
911 the listener destination for each event reported through any of the indications covered by the indication
912 gate referenced by the subscription.

913 **6.4.2 Overlapping coverages of subscriptions**

914 This profile does not specify any rules prohibiting that a listener simultaneously is subscribed to several
915 indication gates with overlapping coverages.

916 For example, a listener could simultaneously be subscribed to a filter collection and to an indication filter
917 contained by that filter collection. As another example, a listener could simultaneously be subscribed to
918 two or more unrelated indication filters that are defined in the same or in different profiles and where the
919 coverages as defined by respective query statements overlap.

920 If separate subscriptions to indication gates with overlapping coverages exist, indications are
921 independently delivered for each individual subscription. This can result in multiple indications being
922 delivered to the listener for the same event. The semantical requirements pertaining to the delivery of
923 indications to subscribed listener destinations are detailed in 7.3.23.2 and 7.3.25.2.

924 **6.4.3 Subscription management authorization**

925 This profile makes no explicit provisions for managing the permissions of a client with respect to its ability
926 to create, modify, or delete subscriptions. Any coordination between clients, or between a client and
927 access management, to govern the ability of one client to make changes that affect the delivery of
928 indications delivered to a listener is outside the scope of this profile.

929 **6.4.4 Listeners**

930 A listener is a WBEM listener that implements applicable portions of this profile. Listeners can be
931 subscribed at an implementation for the delivery of specific sets of indications as exposed by indication
932 gates within that implementation. After a subscription is established within an implementation, indications
933 are delivered to subscribed listeners as respective events occur, and the listeners need to receive and
934 process these indications.

935 In general, a listener is different from the client that establishes its representation within the
936 implementation in the form of a respective listener destination (see 6.4.5); however, clients that also
937 implement listener functionality can establish themselves as listeners.

938 **6.4.5 Listener destinations**

939 A listener destination is an entity that maintains a reference to a listener within an implementation,
940 including information about the protocol applicable to contact the listener; for details, see 7.3.23.

941 A free listener destination is a listener destination that does not currently reference a listener. Clients are
942 enabled to establish a reference to a particular listener; for details, see 7.3.23.3.6.

943 The implementation is responsible for delivering the indications that are passed from any indication gate
944 to any listener referenced by a listener destination that is subscribed to that indication gate. The
945 semantical requirements pertaining to the delivery of indications to subscribed listener destinations are
946 detailed in 7.3.23.2 and 7.3.25.2.

947 Implementations provide functionality enabling clients to control the lifecycle of listener destinations (for
948 example, their creation and destruction), or provide a set of predefined listener destinations along with
949 functionality enabling clients to modify these to refer to different listeners, or provide a combination of
950 both approaches.

951 The second approach requiring the modification of predefined listener destinations is inherently unsafe
952 because activities of different clients can overlap, and race conditions can occur; for that reason the
953 create/delete based approach should be favored.

954 **6.5 Indication service and implementation**

955 **6.5.1 Implementation**

956 An implementation is the realization of applicable portions of this profile within a WBEM server. Within
957 implementations, the functionality defined in this profile may be divided into common parts and
958 referencing profile related parts; for details, see 7.1.

959 **6.5.2 Indication service**

960 An indication service is a component within an implementation that is responsible for delivering
961 indications to listeners. An indication service manages elements such as listener destinations (see 6.4.3)
962 and subscriptions (see 6.4.1), and it may provide support for reliable indication delivery (see 6.1.5) and
963 for dynamic indication filters (see 6.2.6).

964 **6.6 Indication system and referencing profiles**

965 An indication system is a system that hosts a WBEM server with one or more indication services.

966 NOTE The current version of this profile allows only one indication service per indication system; the limitation
967 may be raised in a future version of this profile.

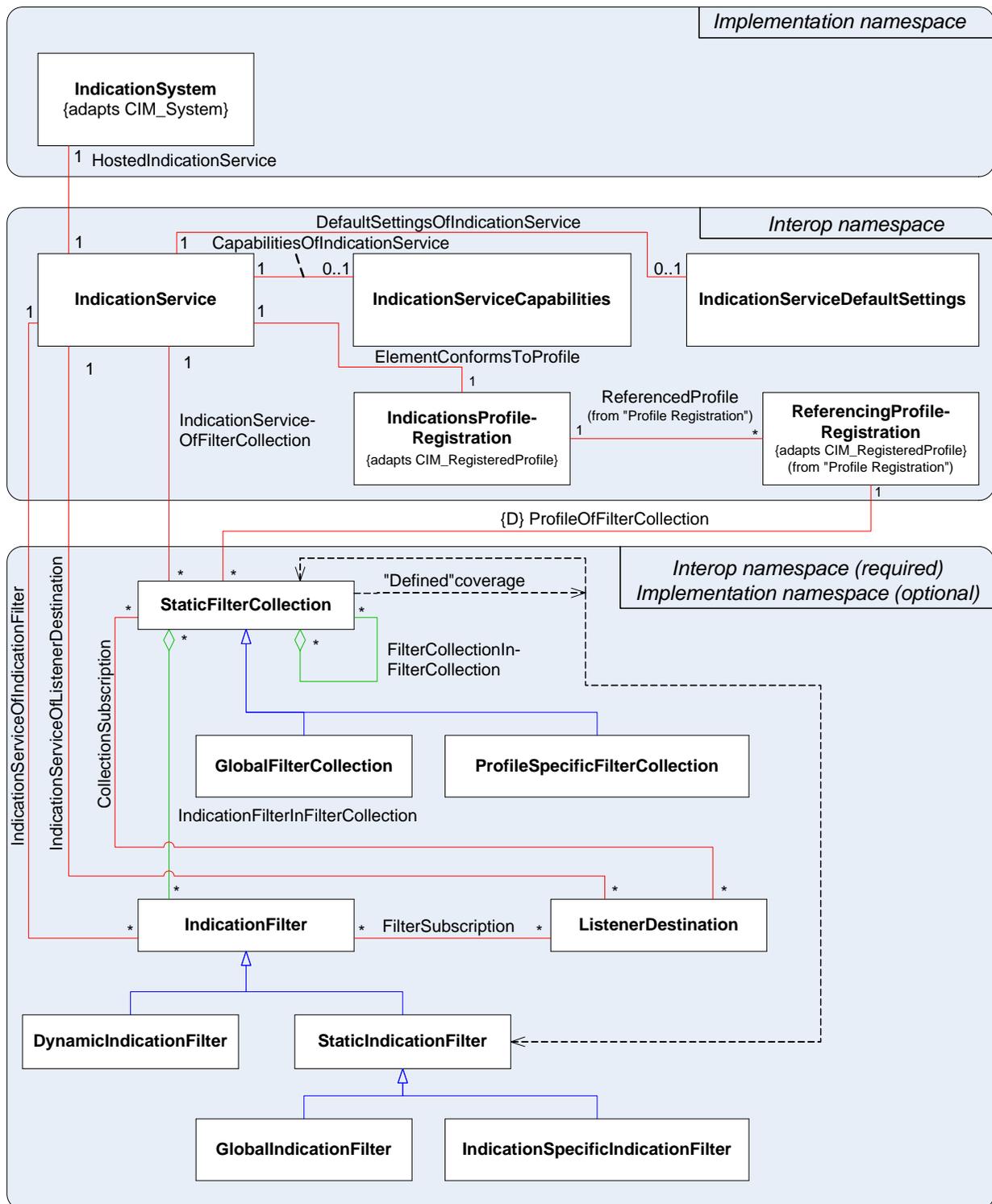
968 In the general case, the scoping systems of referencing profiles are different from the indication system,
969 that is, they are different from the system hosting the WBEM server. In other words, referencing profiles
970 are not required to provide the scope for the indication service, and the central class adaptation of a
971 referencing profile is not required to model the system that hosts the indication service. For that reason,
972 this profile requires that the central class profile advertisement methodology as defined in [DSP1033](#) is
973 applied for advertising this profile; for details, see 7.3.6.

974 For example, consider an Example Fan profile that defines a central Fan adaptation of the CIM_Fan class
975 modeling fans and also defines indications reporting events related to fans and their related elements; in
976 this case the systems containing the fans are not required to be indication systems; particularly, they are
977 not required to host an indication service.

978 As a second example, consider an Example Virtual System profile that defines a central VirtualSystem
979 adaptation of the CIM_ComputerSystem class modeling virtual systems and also defines indications
980 reporting events related to virtual systems and their components; again, the virtual systems are not
981 required to be indication systems, that is, they are not required to host an indication service.

982 **6.7 CIM model**

983 Figure 2 shows the DMTF adaptation diagram for this profile.



984

985

Figure 2 – Indications Profile: DMTF class adaptation diagram

986 The most essential adaptations defined in this profile are listed below, along with their modeled managed
987 object types:

- 988 • the IndicationService adaptation (see 7.3.2) models indication services as described in 6.5.2
- 989 • the IndicationFilter adaptation (see 7.3.11) models indication filters as described in 6.2
- 990 • the StaticFilterCollection adaptation (see 7.3.17) models static filter collections as described
991 in 6.3
- 992 • the StaticIndicationFilter adaptation (see 7.3.17) models static indication filters as described
993 in 6.2.3
- 994 • the ListenerDestination adaptation (see 7.3.23) models listener destinations as described
995 in 6.4.3
- 996 • the AbstractSubscription adaptation (see 7.3.25) models subscriptions as described in 6.4.1

997 Instances of most of these adaptations are instantiated in the Interop namespace; the use of the Interop
998 namespace (see [DSP1033](#)) makes it easier for clients to detect the CIM representations of respective
999 managed objects.

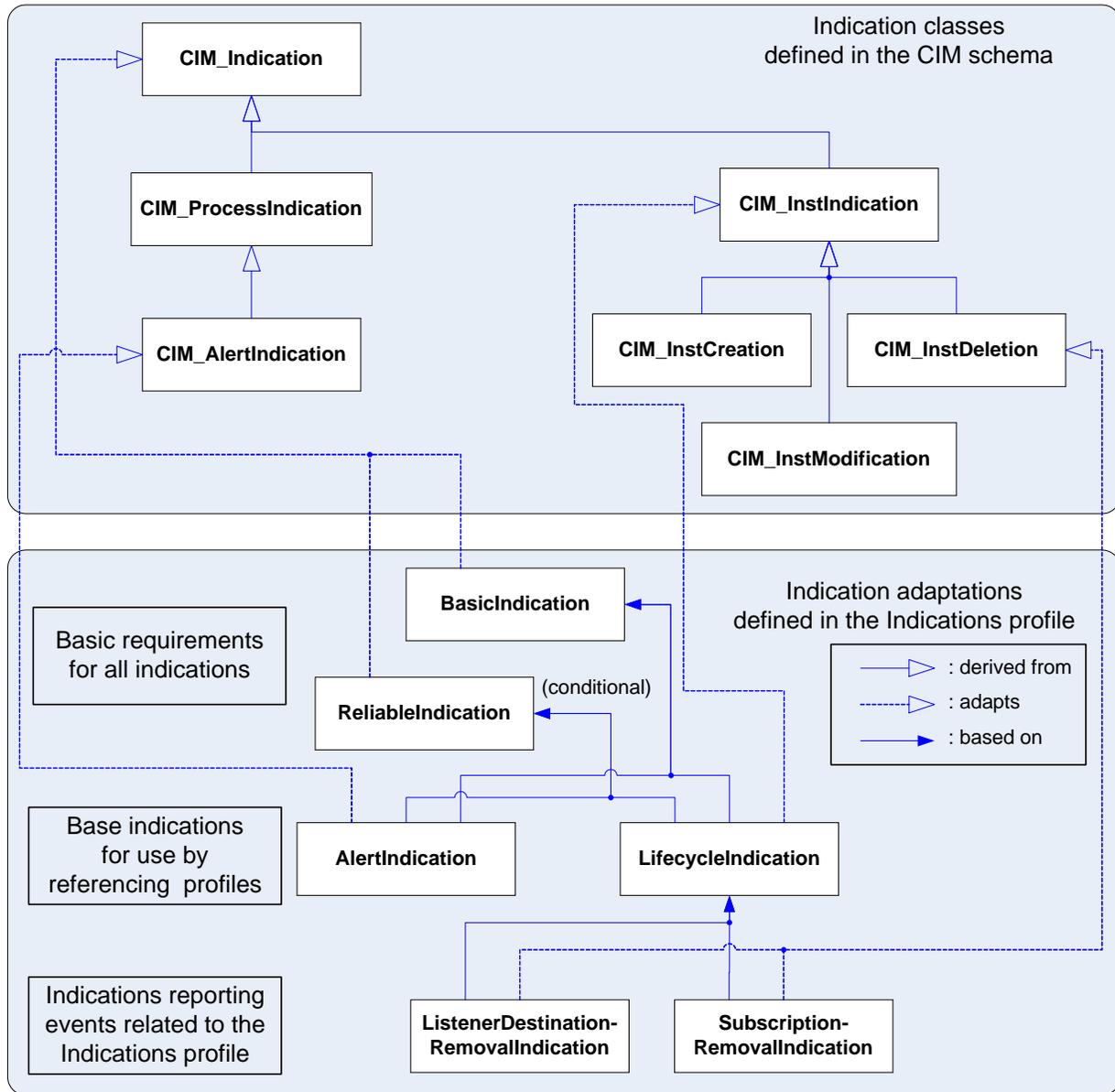
1000 **DEPRECATED**

1001 The ProfileOfFilterCollection association adaptation models the relationship between filter collections and
1002 the registration of this profile.

1003 NOTE The ProfileOfFilterCollection association adaptation (defined as the CIM_ConcreteDependency "profile
1004 class" in version 1.1 of this profile) is deprecated in version 1.2 of this profile in favor of a naming
1005 convention for static filter collections that enables their unique identification.

1006 **DEPRECATED**

1007 Figure 3 depicts the adaptations of indication classes defined by this profile along with the adapted
 1008 indication classes.



1009

1010 **Figure 3 – Indications Profile: Indication adaptations and adapted indication classes**

1011 The most essential indication adaptations defined in this profile are listed below, along with their modeled
 1012 indications:

- 1013 • the BasicIndication adaptation (see 7.3.29) models indications as described in 6.1.2
- 1014 • the ReliableIndication adaptation (see 7.3.30) models reliable indications as described in 6.1.5;
 1015 this adaptation specifies additional optional requirements that can be implemented separately
 1016 from the requirements of other indication adaptations.

- 1017 • the AlertIndication adaptation (see 7.3.31) models alert indication as described in 6.1.2.2; it is
1018 an abstract adaptation available to referencing profiles in order to define their own alert
1019 indications
- 1020 • the LifecycleIndication adaptation (see 7.3.32) models lifecycle indications as described
1021 in 6.1.2.3; it is an abstract adaptation available to referencing profiles in order to define their
1022 own lifecycle indications.

1023 7 Implementation

1024 7.1 Separation of requirements

1025 This profile defines implementation requirements for implementations (for example, WBEM servers
1026 implementing this profile) and for listeners (for example, WBEM listeners implementing this profile).

1027 The implementation requirements for implementations are further separated into WBEM server related
1028 requirements and referencing profile related requirements, as follows:

- 1029 • Requirements that address the infrastructure for the delivery of indications (including the
1030 management of listener destinations and subscriptions) are WBEM server related requirements,
1031 and are typically implemented only once within an implementation.
- 1032 • Requirements that address the generation of indications are related to the referencing profile
1033 defining those indications, and are typically implemented as part of the implementation of that
1034 referencing profile.
- 1035 • Requirements that address functionality related to indication filters and filter collections are
1036 referencing profile related requirements.

1037 However, WBEM servers may contain other facilities allowing implementations of referencing
1038 profiles to delegate some of their implementation responsibilities to these facilities. For example,
1039 within WBEM servers providing a CIM instance repository the implementations of referencing
1040 profiles can delegate storing indication filters and filter collections to the CIM instance
1041 repository, such that in this case the implementation requirements for referencing profiles are
1042 effectively reduced to storing respective objects into the repository when the implementation of
1043 the referencing profile is installed.

1044 In this profile WBEM server related implementation requirements are marked with a phrase such as the
1045 following:

1046 "The requirements in this subclause are WBEM server related implementation requirements."

1047 In this profile referencing profile related implementation requirements are marked with a phrase such as
1048 the following:

1049 "The requirements in this subclause are referencing profile related implementation requirements."

1050 This facilitates explicit distinction of WBEM server related implementation requirements as opposed to
1051 requirements related to the implementation of referencing profiles.

1052 7.2 Features

1053 7.2.1 DynamicIndicationFilters

1054 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1055 The implementation of the DynamicIndicationFilters feature provides functionality for dynamic indication
1056 filters; for a description of dynamic indication filters, see 6.2.6.

1057 The granularity of the DynamicIndicationFilters feature is per IndicationService instance (see 7.3.2).

1058 The requirement level of the DynamicIndicationFilters feature is optional.

1059 The implementation of the DynamicIndicationFilters feature for a particular IndicationService instance is
1060 indicated by a value of True for the FilterCreationEnabled property.

1061 **7.2.2 IndicationServiceInitialSettingsExposed**

1062 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1063 The implementation of the IndicationServiceInitialSettingsExposed feature provides information about the
1064 initial settings of an indication service.

1065 The granularity of the IndicationServiceInitialSettingsExposed feature is per
1066 IndicationService instance (see 7.3.2).

1067 The requirement level of the IndicationServiceInitialSettingsExposed feature is optional.

1068 The availability of the IndicationServiceInitialSettingsExposed feature for a particular IndicationService
1069 instance is indicated by the presence of an IndicationServiceInitialSettings instance (see 7.3.9)
1070 associated through an InitialSettingsOfIndicationService instance (see 7.3.10).

1071 **7.2.3 IndicationServiceModification**

1072 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1073 The implementation of the IndicationServiceModification feature provides functionality for client requested
1074 dynamic modification of an indication service.

1075 The granularity of the IndicationServiceModification feature is per IndicationService instance (see 7.3.2).

1076 The requirement level of the IndicationServiceModification feature is optional.

1077 The availability of the IndicationServiceModification feature for a particular IndicationService instance is
1078 indicated if an IndicationServiceCapabilities (see 7.3.7) instance representing the capabilities of the
1079 represented indication service exists and is associated via the CapabilitiesOfIndicationService association
1080 (see 7.3.8), and in that instance the value True is set for any of the following properties:
1081 FilterCreationEnabledIsSettable, DeliveryRetryAttemptsIsSettable, DeliveryRetryIntervalsSettable,
1082 SubscriptionRemovalActionIsSettable, or SubscriptionRemovalTimeIntervalsSettable.

1083 **7.2.4 ReliableIndications**

1084 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1085 The implementation of the ReliableIndications feature provides functionality for reliable indications as
1086 described in 6.1.5. For further details, see 7.3.30 and 7.4.

1087 The granularity of the ReliableIndications feature is per IndicationService instance (see 7.3.2).

1088 The requirement level of the ReliableIndications feature is optional. The implementation of the
1089 ReliableIndications feature is also optional for listeners; in this case, the granularity is once per listener,
1090 and the discovery mechanism does not apply.

1091 The availability of the ReliableIndications feature for a particular IndicationService instance is indicated by
1092 a value larger than 0 for the DeliveryRetryAttempts property.

1093 **7.2.5 SuppressRepeatNotificationPolicy**

1094 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1095 The implementation of the SuppressRepeatNotificationPolicy feature provides functionality for
1096 suppressing repeated indication delivery by implementing the "suppress repeated indication delivery
1097 avoidance policy", as described in 6.1.6.3.

1098 The granularity of the SuppressRepeatNotificationPolicy feature is per implementation.

1099 The requirement level of the SuppressRepeatNotificationPolicy feature is optional.

1100 The availability of the SuppressRepeatNotificationPolicy feature is indicated by the value 3 (Suppress) for
1101 the RepeatNotificationPolicy property in AbstractSubscription instances (see 7.3.25) representing existing
1102 subscriptions.

1103 NOTE The discovery mechanism specified here is only rudimentary because the feature presence can only be
1104 discovered if at least one exploiting subscription is discovered. A future version of this profile is expected
1105 to introduce a new property into the CIM_IndicationServiceCapabilities class that indicates the presence of
1106 the feature per indication service.

1107 **7.2.6 DelayRepeatNotificationPolicy**

1108 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1109 The implementation of the DelayRepeatNotificationPolicy feature provides functionality for suppressing
1110 repeated indication delivery by implementing the "delayed indication delivery avoidance policy", as
1111 described in 6.1.6.4.

1112 The granularity of the DelayRepeatNotificationPolicy feature is per implementation.

1113 The requirement level of the DelayRepeatNotificationPolicy feature is optional.

1114 The availability of the DelayRepeatNotificationPolicy feature is indicated by the value 4 (Delay) for the
1115 RepeatNotificationPolicy property in AbstractSubscription instances (see 7.3.25) representing existing
1116 subscriptions.

1117 NOTE The discovery mechanism specified here is only rudimentary because the feature presence can only be
1118 discovered if at least one exploiting subscription is discovered. A future version of this profile is expected
1119 to introduce a new property into the CIM_IndicationServiceCapabilities class that indicates the presence of
1120 the feature per indication service.

1121 **7.2.7 IndividualFilterSubscription**

1122 The implementation of the IndividualFilterSubscription feature provides functionality for subscriptions to
1123 individual indication filters.

1124 The granularity of the IndividualFilterSubscription feature is per IndicationFilter instance (see 7.3.11).

1125 The requirement level of the IndividualFilterSubscription feature is optional.

1126 The availability of the IndividualFilterSubscription feature for a particular IndicationFilter instance is
1127 indicated by the value True for the IndividualSubscriptionSupported property.

1128 **7.2.8 FilterCollectionCoverageExposure**

1129 The implementation of the FilterCollectionCoverageExposure feature provides functionality for exposing
1130 the coverage of static filter collections.

1131 The granularity of the FilterCollectionCoverageExposure feature is per
1132 StaticFilterCollection instance (see 7.3.17).

1133 The requirement level of the FilterCollectionCoverageExposure feature is optional.

1134 The availability of the FilterCollectionCoverageExposure feature for a particular StaticFilterCollection
1135 instance is indicated through at least one instance of either the IndicationFilterInFilterCollection
1136 association adaptation (see 7.3.19) or the FilterCollectionInFilterCollection association adaptation (see
1137 7.3.20) referencing the StaticFilterCollection instance.

1138 **7.3 Adaptations**

1139 **7.3.1 Conventions**

1140 This profile repeats the effective values of certain Boolean qualifiers as part of property requirements, or
1141 of method parameter requirements. The following convention is established: If the name of a qualifier is
1142 listed, its effective value is True; if the qualifier name is not listed, its effective value is False. The
1143 convention is applied in the following cases:

- 1144 • In: indicates that the parameter is an input parameter
- 1145 • Out: indicates that the parameter is an output parameter
- 1146 • Key: indicates that the property is a key (that is, its value is part of the instance part)
- 1147 • Required: indicates that the element value shall be non-Null

1148 This profile defines operation requirements based on [DSP0223](#).

1149 For adaptations of ordinary classes and of associations the implementation requirements for operations
1150 are specified in adaptation-specific subclauses of 7.3.

1151 **7.3.2 IndicationService: CIM_IndicationService**

1152 **7.3.2.1 General**

1153 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1154 The IndicationService adaptation models indication services; indication services are described in 6.5.2.

1155 The implementation type of the IndicationService adaptation is: "instantiated".

1156 The IndicationService adaptation shall conform to the requirements for "central classes" defined in the
1157 Profile Registration profile; for details, see [DSP1033](#).

1158 **7.3.2.2 Initial behavior**

1159 If the IndicationServiceInitialSettingsExposed feature (see 7.2.2) is implemented, the initial behavior of an
1160 indication service shall be as exposed by the IndicationServiceInitialSettings instance (see 7.3.9) that is
1161 associated with the IndicationService instance representing that indication service through an
1162 InitialSettingsOfIndicationService instance (see 7.3.10).

1163 If the IndicationServiceInitialSettingsExposed feature (see 7.2.2) is not implemented, then the initial
1164 behavior of the indication service shall be as follows:

- 1165 • Retry the delivery of an indication after a delivery failure three additional times, each time
1166 waiting 20 seconds before the retry, and indicate this behavior with a value of 3 for the
1167 DeliveryRetryAttempts property (see 7.3.2.3.3) and the value 20 for the DeliveryRetryInterval
1168 property (see 7.3.2.3.4) in the IndicationService instance representing the indication service

- 1169 • Remove affected subscriptions after 30 days, and indicate this behavior with a value of 2
 - 1170 (Remove) for the SubscriptionRemovalAction property (see 7.3.2.3.5), and a value of 2,592,000
 - 1171 seconds (30 days) for the SubscriptionRemovalTimeInterval property (see 7.3.2.3.6) in the
 - 1172 IndicationService instance representing the indication service
- 1173 NOTE With respect to the availability of DynamicIndicationFilters feature (see 7.2.1) as indicated by the value of
- 1174 the FilterCreationEnabled property an recommended initial behavior is not established; instead the
- 1175 implementation is required to always expose the available behavior; see 7.3.2.3.2.

1176 **7.3.2.3 Element requirements**

1177 **7.3.2.3.1 General**

1178 Table 4 lists the element requirements for the IndicationService adaptation.

1179 **Table 4 – IndicationService: Element requirements**

Elements	Requirement	Description
Properties		
Name	Mandatory	Key: See CIM schema definition.
CreationClassName	Mandatory	Key: See CIM schema definition.
SystemName	Mandatory	Key: See CIM schema definition.
SystemCreationClassName	Mandatory	Key: See CIM schema definition.
FilterCreationEnabled	Mandatory	See 7.3.2.3.2.
DeliveryRetryAttempts	Mandatory	See 7.3.2.3.3.
DeliveryRetryInterval	Mandatory	See 7.3.2.3.4.
SubscriptionRemovalAction	Mandatory	See 7.3.2.3.5.
SubscriptionRemovalTimeInterval	Mandatory	See 7.3.2.3.6.
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .
ModifyInstance()	Conditional	See 7.3.2.3.7 and DSP0223 .

1180 If the ModifyInstance() operation is implemented (see 7.3.2.3.7), the values of some properties might be

1181 modifiable through client requests; see 7.3.7 for details on indicating those properties whose values are

1182 actually modifiable.

1183 **7.3.2.3.2 Property: FilterCreationEnabled**

1184 The value of the FilterCreationEnabled property shall reflect whether the DynamicIndicationFilters feature

1185 (see 7.2.1) is available for the IndicationService instance. A value of False indicates that the feature is not

1186 available; a value of True indicates that the feature is available.

1187 **7.3.2.3.3 Property: DeliveryRetryAttempts**

1188 The value of the DeliveryRetryAttempts property shall reflect the number of times that the implementation
 1189 is going to retry the delivery of an indication to a particular listener in the case of delivery failures. This
 1190 value does not include the initial delivery attempt.

1191 A value larger than 0 indicates that the ReliableIndications feature (see 7.2.4) is available. The value 0
 1192 indicates that the ReliableIndications feature is not available.

1193 **7.3.2.3.4 Property: DeliveryRetryInterval**

1194 The value of the DeliveryRetryInterval property shall reflect the minimal time interval in seconds that the
 1195 implementation waits before delivering an indication to a particular listener destination after a previous
 1196 delivery failure.

1197 **7.3.2.3.5 Property: SubscriptionRemovalAction**

1198 The value of the SubscriptionRemovalAction property shall reflect the removal action for subscriptions
 1199 after two failed indication deliveries where the time interval between the failed deliveries, without any
 1200 intermediate successful indication delivery, exceeds the timeout reflected by the value of the
 1201 SubscriptionRemovalTimeInterval property.

1202 **7.3.2.3.6 Property: SubscriptionRemovalTimeInterval**

1203 The value of the SubscriptionRemovalTimeInterval property shall reflect the minimum time interval that
 1204 implementations shall wait after two failed indication deliveries without any intermediate successful
 1205 indication delivery, before performing the activity designated by the value of the
 1206 SubscriptionRemovalAction property.

1207 **7.3.2.3.7 Method: ModifyInstance()**

1208 The implementation of the ModifyInstance() operation enables clients to modify aspects of the behavior
 1209 of the represented indication service.

1210 The requirement level of the ModifyInstance() operation is conditional.

1211 Condition: The IndicationServiceModification feature is implemented; for a description, see 7.2.3.

1212 Information about which properties are modifiable is provided by an IndicationServiceCapabilities
 1213 instance that is associated to the IndicationService instance representing the indication service; see 7.3.7
 1214 and 7.3.8.

1215 Table 5 lists the error reporting requirements for the ModifyInstance() operation on IndicationService
 1216 instances. If any of the error situations described in the Description column of Table 5 matches, the
 1217 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1218 reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1219 **Table 5 – ModifyInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the behavior requested by the value of the FilterCreationEnabled property in the input IndicationService instance, as described in 7.3.2.3.2.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the behavior requested by the value of the DeliveryRetryAttempts property in the input IndicationService instance, as described in 7.3.2.3.3.

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the delivery retry interval requested by the value of the DeliveryRetryInterval property, as described in 7.3.2.3.4.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the subscription removal action requested by the value of the SubscriptionRemovalAction property in the input IndicationService instance, as described in 7.3.2.3.5.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the subscription removal time interval requested by the value of the SubscriptionRemovalTimeInterval property in the input IndicationService instance, as described in 7.3.2.3.6.
CIM_ERR_NOT_SUPPORTED	Mandatory	The IndicationServiceModification feature is not implemented; see 7.2.3 and 7.3.7.
CIM_ERR_FAILED	Mandatory	The IndicationServiceModification feature is not available for the IndicationService instance; see 7.2.3 and 7.3.7.

1220 If the ModifyInstance() operation is successful, the requested modification on the indication service shall
 1221 be applied, and — as a consequence — shall be reflected in all IndicationService instances that
 1222 represent the modified indication service and are exposed by the implementation.

1223 If the ModifyInstance() operation fails, the requested modification on the indication service shall not be
 1224 applied, and — as a consequence — all IndicationService instances that represent the indication service
 1225 shall remain unchanged.

1226 **7.3.2.4 Instance requirements**

1227 Within an implementation there shall be exactly one indication service. That indication service shall be
 1228 represented by an IndicationService instance in the Interop namespace.

1229 NOTE 1 The reasons for requiring exactly one indication service are a) other elements defined in this profile (such
 1230 as subscriptions, listener destinations, or dynamic indication filters) require a relationship to the indication
 1231 service, and b) the modeled use of the CreateInstance() operation does not provide for expressing that
 1232 required relationship at creation time. For these reasons an indication service must be implied at creation
 1233 time, and the simplest approach for that is allowing just one indication service. Future versions of this
 1234 profile might lift the single instance restriction, for example by modeling respective creation methods with
 1235 parameters that enable establishing the required relationship to a specifiable indication service.

1236 NOTE 2 In some places in this profile multiple indication services are mentioned. This is not meant to lift the
 1237 restriction established in this subclause, but to accommodate the future introduction of multiple indication
 1238 services.

1239 **7.3.3 IndicationSystem: CIM_System**

1240 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1241 The IndicationSystem adaptation models indication systems; indication systems are described in 6.6.

1242 The implementation type of the IndicationSystem adaptation is: "instantiated".

1243 The IndicationSystem adaptation shall conform to the requirements for "scoping classes" defined in the
 1244 Profile Registration profile; for details, see [DSP1033](#).

1245 Table 6 lists the element requirements of the IndicationSystem adaptation.

1246 **Table 6 – IndicationSystem: Element requirements**

Elements	Requirement	Description
Properties		
Name	Mandatory	Key: See CIM schema definition.
CreationClassName	Mandatory	Key: See CIM schema definition.
Operations		
Associators()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .

1247 **7.3.4 HostedIndicationService: CIM_HostedService**

1248 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1249 The HostedIndicationService adaptation models the relationship between an indication service and its
 1250 hosting indication system.

1251 The implementation type of the HostedIndicationService association adaptation is: "instantiated".

1252 Table 7 lists the element requirements for the HostedIndicationService association adaptation.

1253 **Table 7 – HostedIndicationService: Element requirements**

Elements	Requirement	Description
Properties		
Antecedent	Mandatory	Key: Value shall reference the IndicationSystem instance Multiplicity: 1
Dependent	Mandatory	Key: Value shall reference the IndicationService instance Multiplicity: 1
Operations		
GetInstance()	Mandatory	See DSP0223 .

Elements	Requirement	Description
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1254 Each IndicationSystem instance (see 7.3.3) shall be associated through a HostedIndicationService
 1255 instance with the IndicationService instance (see 7.3.2) representing the indication service hosted by the
 1256 indication system represented by the IndicationSystem instance.

1257 **7.3.5 IndicationsProfileRegistration: CIM_RegisteredProfile**

1258 **7.3.5.1 General**

1259 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1260 The IndicationsProfileRegistration adaptation models the profile registration of this profile, that is, the
 1261 representation of the specific implemented version 1.2.0 of this profile.

1262 The implementation type of the IndicationsProfileRegistration adaptation is: "instantiated".

1263 The specific implemented version of this profile shall be represented by IndicationsProfileRegistration
 1264 instances in the Interop namespace.

1265 NOTE The existence of an instance of this adaptation indicates that version 1.2.0 of this profile is implemented at
 1266 least once within the WBEM server.

1267 Table 8 lists the element requirements for the IndicationsProfileRegistration adaptation.

1268 **Table 8 – IndicationsProfileRegistration: Element requirements**

Elements	Requirement	Description
Base adaptations		
ProfileRegistration::CIM_RegisteredProfile		The IndicationsProfileRegistration adaptation shall conform to the requirements for the CIM_RegisteredProfile "profile class" defined in the Profile Registration profile; see DSP1033 .
Properties		
InstanceID	Mandatory	Key: See CIM schema definition.
RegisteredName	Mandatory	Value shall be "Indications".
RegisteredVersion	Mandatory	Value shall be "1.2.0".
RegisteredOrganization	Mandatory	Value shall be 2 (DMTF).

1269 NOTE Operation requirements are defined by the base "profile class" CIM_RegisteredProfile defined in
 1270 [DSP1033](#).

1271 **7.3.6 ElementConformsToProfile: CIM_ElementConformsToProfile**

1272 The ElementConformsToProfile adaptation models the relationship between an indication service and the
 1273 profile registration of this profile (see 7.3.5).

1274 The implementation type of the ElementConformsToProfile association adaptation is: "instantiated".

1275 Table 9 lists the element requirements for the ElementConformsToProfile association adaptation.

1276 **Table 9 – ElementConformsToProfile: Element requirements**

Elements	Requirement	Description
Base adaptations		
Profile Registration::CIM_Element-ConformsToProfile	Mandatory	The ElementConformsToProfile association adaptation shall conform to the requirements for the CIM_ElementConformsToProfile "profile class" defined in the Profile Registration profile; see DSP1033 .
Properties		
ConformantStandard	Mandatory	Key: Value shall reference the IndicationsProfileRegistration instance Multiplicity: 1
ManagedElement	Mandatory	Key: Value shall reference the IndicationService instance. Multiplicity: 1
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1277 Each IndicationService instance (see 7.3.2) shall be associated through an ElementConformsToProfile
1278 instance with an IndicationsProfileRegistration instance (see 7.3.5).

1279 NOTE By requiring the implementation of the ElementConformsToProfile adaptation, this profile in fact requires
1280 the central class profile advertisement methodology defined in [DSP1033](#). The scoping class profile
1281 advertisement methodology is not applicable because the central instances of implementations of
1282 referencing profiles will in almost all cases not be identical with the central instance of this profile, that is,
1283 the IndicationSystem instance required by 7.3.3. Note that this does not restrict referencing profiles from
1284 choosing a different methodology for their profile advertisement.

1285 **7.3.7 IndicationServiceCapabilities: CIM_IndicationServiceCapabilities**

1286 **7.3.7.1 General**

1287 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1288 The IndicationServiceCapabilities adaptation models the capabilities of indication services; indication
1289 services are described in 6.5.2.

1290 The requirement level of the IndicationServiceCapabilities adaptation is conditional.

1291 Condition: The IndicationServiceModification feature is implemented; see 7.2.3.

1292 The implementation type of the IndicationServiceCapabilities adaptation is: "instantiated".

1293 **7.3.7.2 Element requirements**

1294 **7.3.7.2.1 General**

1295 Table 10 lists the element requirements for the IndicationServiceCapabilities adaptation.

1296 **Table 10 – IndicationServiceCapabilities: Element requirements**

Element	Requirement	Description
Properties		
InstanceID	Mandatory	Key: See CIM schema definition.
FilterCreationEnabledIsSettable	Mandatory	See 7.3.7.2.2
DeliveryRetryAttemptsIsSettable	Mandatory	Value shall indicate whether the implementation supports modification of the DeliveryRetryAttempts property of the associated IndicationService instance
DeliveryRetryIntervalsSettable	Mandatory	Value shall indicate whether the implementation supports modification of the DeliveryRetryInterval property of the associated IndicationService instance
SubscriptionRemovalActionIsSettable	Mandatory	Value shall indicate whether the implementation supports modification of the SubscriptionRemovalAction property of the associated IndicationService instance
SubscriptionRemovalTimeIntervals-Settable	Mandatory	Value shall indicate whether the implementation supports modification of the SubscriptionRemovalTimeInterval property of the associated IndicationService instance
MaxListenerDestinations	Mandatory	Value shall indicate the maximum number of listener destinations
MaxActiveSubscriptions	Mandatory	Value shall indicate the maximum number of active subscriptions
SubscriptionsPersisted	Mandatory	Value shall indicate whether subscriptions are persisted across restarts of the indication service
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .

1297 **7.3.7.2.2 Property: FilterCreationEnabledIsSettable**

1298 **DEPRECATED**

1299 The value of the FilterCreationEnabledIsSettable property shall indicate whether the implementation
 1300 supports modification of the FilterCreationEnabled property of the associated IndicationService instance.

1301 NOTE Values other than False are deprecated because it does not make sense enabling clients to set values of
 1302 properties that represent functionality that is either implemented or not implemented.

1303 **DEPRECATED**

1304 The value of the FilterCreationEnabledIsSettable property should be False, indicating that the
 1305 implementation does not support the modification of the FilterCreationEnabled property of the associated
 1306 IndicationService instance.

1307 **7.3.8 CapabilitiesOfIndicationService: CIM_ElementCapabilities**

1308 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.
 1309 The CapabilitiesOfIndicationService adaptation models the relationship between an indication service and
 1310 its capabilities.
 1311 The requirement level of the CapabilitiesOfIndicationService adaptation is conditional.
 1312 Condition: The IndicationServiceModification feature is implemented; see 7.2.3.
 1313 The implementation type of the CapabilitiesOfIndicationService association adaptation is: "instantiated".
 1314 Table 11 lists the element requirements for the CapabilitiesOfIndicationService association adaptation.

1315 **Table 11 – CapabilitiesOfIndicationService: Element requirements**

Elements	Requirement	Description
Properties		
ManagedElement	Mandatory	Key: Value shall reference the IndicationService instance Multiplicity: 1
Capabilities	Mandatory	Key: Value shall reference the IndicationServiceCapabilities instance Multiplicity: 0..1
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1316 Each IndicationService instance (see 7.3.2) shall be associated through a CapabilitiesOfIndicationService
 1317 instance with at most one IndicationServiceCapabilities instance (see 7.3.7) representing the capabilities
 1318 of the indication service represented by the IndicationService instance.

1319 **7.3.9 IndicationServiceInitialSettings: CIM_IndicationServiceSettingData**

1320 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.
 1321 The IndicationServiceInitialSettings adaptation models initial settings for indication services; indication
 1322 services are described in 6.5.2. The initial settings of an indication service are the settings that apply at
 1323 the point in time when the WBEM server hosting the indication service initially starts up the indication
 1324 service.
 1325 The requirement level of the IndicationServiceInitialSettings adaptation is conditional.
 1326 Condition: The IndicationServiceInitialSettingsExposed feature is implemented; see 7.2.2.
 1327 The implementation type of the IndicationServiceInitialSettings adaptation is: "instantiated".
 1328 Table 12 lists the element requirements for the IndicationServiceInitialSettings adaptation.

1329

Table 12 – IndicationServiceInitialSettings: Element requirements

Elements	Requirement	Description
Properties		
InstanceID	Mandatory	Key: See CIM schema definition.
FilterCreationEnabled	Mandatory	Value shall be the initial value for the FilterCreationEnabled property in the associated IndicationService instance; the requirements of 7.3.2.3.3 apply.
DeliveryRetryAttempts	Mandatory	Value shall be the initial value for the DeliveryRetryAttempts property in the associated IndicationService instance; the requirements of 7.3.2.3.4 apply.
SubscriptionRemovalAction	Mandatory	Value shall be the initial value for the SubscriptionRemovalAction property in the associated IndicationService instance; the requirements of 7.3.2.3.5 apply.
SubscriptionRemovalTimeInterval	Mandatory	Value shall be the initial value for the SubscriptionRemovalTimeInterval property in the associated IndicationService instance; the requirements of 7.3.2.3.5 apply.
SubscriptionRemovalTimeInterval	Mandatory	Value shall be the initial value for the SubscriptionRemovalTimeInterval property (see 7.3.2.3.6) in the associated IndicationService instance
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .

1330 The initial settings of an indication service shall be represented by an IndicationServiceInitialSettings
 1331 instance in the Interop namespace.

1332 **7.3.10 InitialSettingsOfIndicationService: CIM_ElementSettingData**

1333 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1334 The InitialSettingsOfIndicationService association adaptation models the relationship between an
 1335 indication service and its initial settings; indication services are described in 6.5.2.

1336 The requirement level of the InitialSettingsOfIndicationService association adaptation is conditional.

1337 Condition: The IndicationServiceInitialSettingsExposed feature is implemented; see 7.2.2.

1338 The implementation type of the InitialSettingsOfIndicationService association adaptation is: "instantiated".

1339 Table 13 lists the element requirements for the InitialSettingsOfIndicationService association adaptation.

1340

Table 13 – InitialSettingsOfIndicationService: Element requirements

Elements	Requirement	Description
Properties		
ManagedElement	Mandatory	Key: Value shall reference an IndicationService instance Multiplicity: 1
SettingData	Mandatory	Key: Value shall reference the IndicationServiceInitialSettings instance Multiplicity: 0..1
IsDefault	Mandatory	Value shall be 1 (Is Default)
IsNext	Mandatory	Value shall be 1 (Is Next)
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1341 Each IndicationService instance (see 7.3.2) shall be associated through a
 1342 InitialSettingsOfIndicationService instance with at most one IndicationServiceInitialSettings instance (see
 1343 7.3.9) representing the initial settings of the indication service represented by the IndicationService
 1344 instance.

1345 **7.3.11 IndicationFilter: CIM_IndicationFilter**

1346 **7.3.11.1 General**

1347 The requirements in this subclause are referencing profile and WBEM server related implementation
 1348 requirements.

1349 The IndicationFilter adaptation models indication filters; indication filters are described in 6.2.

1350 The implementation type of the IndicationFilter adaptation is: "abstract".

1351 **7.3.11.2 Semantical requirements**

1352 For a particular indication filter the implementation shall filter any indication generated by (indication-
 1353 specific parts of) the implementation that is within the coverage of the indication filter, that is, that meets
 1354 both of the following requirements:

- 1355 • it matches the query statement (see 7.3.11.3.5) given by the value of the Query property in the
 1356 IndicationFilter instance representing the indication filter
- 1357 • its indication origin (see 6.1.2.4) is one of the local namespaces identified by the value of the
 1358 SourceNamespaces[] array property in that instance, or, in case that value is NULL, is the local
 1359 namespace in which the IndicationFilter instance representing the indication filter resides

1360 For the particular indication filter the implementation shall ignore any generated indication that does not
 1361 meet these requirements.

1362 Indications that passed an indication filter need to be further processed; see the requirements on the
 1363 IndicationFilterName property defined in 7.3.29.4.2, and the semantical requirements on listener
 1364 destinations defined in 7.3.23.2, and on subscriptions defined in 7.3.25.2. If implemented, the
 1365 requirements for reliable indications as defined in 7.3.30 and 7.4 may apply.

1366 Note that the indication filter semantics apply regardless of which profile specified the indications and
 1367 indication filters; thus an indication specified in one referencing profile is required to be considered by
 1368 indication filters specified in that referencing profile, but also by those specified in any other referencing
 1369 profile or in this profile and by those not specified in any profile.

1370 The indication filter semantics defined in this subclause do not require that an implementation implements
 1371 any of the indications within the coverage of an indication filter. However, referencing profiles may define
 1372 additional semantics for indication filters they define, including the case that the existence of a particular
 1373 IndicationFilter instance indicates that one or all indications within the coverage of the represented
 1374 indication filter are implemented. Of course, this approach is only feasible if the coverage covers one or
 1375 just a few indications.

1376 **7.3.11.3 Element requirements**

1377 **7.3.11.3.1 General**

1378 Table 14 lists the element requirements for the IndicationFilter adaptation.

1379 **Table 14 – IndicationFilter: Element requirements**

Elements	Requirement	Description
Properties		
Name	Mandatory	Key: See 7.3.11.3.2.
CreationClassName	Mandatory	Key: See CIM schema definition.
SystemName	Mandatory	Key: See CIM schema definition.
SystemCreationClassName	Mandatory	Key: See CIM schema definition.
SourceNamespaces[]	Mandatory	See 7.3.11.3.3.
IndividualSubscriptionSupported	Mandatory	See 7.3.11.3.4.
Query	Mandatory	See 7.3.11.3.5.
QueryLanguage	Mandatory	See 7.3.11.3.6.
Operations		
Associators()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .

1380 **7.3.11.3.2 Property: Name**

1381 The value of the Name property shall be the name of the indication filter; it shall be formatted as defined
 1382 by the following ABNF rule:

1383 `OrgID ":" RegisteredName ":" UniqueID`

1384 `OrgID` shall identify the business entity owning the referencing profile. `OrgID` shall include a copyrighted,
 1385 trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID
 1386 assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness,
 1387 `OrgID` shall not contain a colon (:). For referencing profiles owned by DMTF, `OrgID` shall match
 1388 "DMTF".

1389 RegisteredName shall be the registered name of the referencing profile, as defined by the value of the
 1390 RegisteredName property in the RegisteredProfile instance representing the implemented version of that
 1391 profile.

1392 UniqueID shall uniquely identify the represented indication filter within the referencing profile.

1393 **DEPRECATED**

1394 For compatibility with version 1.0 of this profile, referencing profiles owned by business entities other than
 1395 DMTF may in addition define values for the Name property that are formatted as defined by the following
 1396 ABNF rule:

1397 OrgID ":" UniqueID

1398 Where:

1399 OrgID is defined above in this subclause.

1400 UniqueID shall uniquely identify the instance within the business entity owning the referencing
 1401 profile.

1402 Version 1.1 of this profile has deprecated this additional format.

1403 **DEPRECATED**

1404 **7.3.11.3.3 Property: SourceNamespaces**

1405 A non-Null value of this property is required for IndicationFilter instances in the Interop namespace; for
 1406 IndicationFilter instances in other namespaces it is optional.

1407 If not Null, the value of the SourceNamespaces[] array property shall contain the names of local
 1408 namespaces that are considered as potential indication origin namespaces (see 6.1.2.4) during indication
 1409 filtering; see 7.3.11.2. The value shall not be an empty array.

1410 It is not required that the local namespaces identified by elements of value of the SourceNamespaces[]
 1411 array property exist. If a non-existing local namespace is identified, no indications can originate out of that
 1412 non-existing namespace; consequently, that element does not have an effect on indication filtering.
 1413 However, if the identified namespace is added to the implementation at a later point in time, per the
 1414 requirements of 7.3.11.2 indications originating out of that namespace are to be considered for indication
 1415 filtering from then on.

1416 The value elements of the SourceNamespaces[] array property shall be formatted using the format that
 1417 the implementation uses for value of the Name property in instances of the CIM_Namespace class that
 1418 represent namespaces.

1419 **7.3.11.3.4 Property: IndividualSubscriptionSupported**

1420 The value of the IndividualSubscriptionSupported property shall be True if the IndividualFilterSubscription
 1421 feature (see 7.2.7) is available for the IndicationFilter instance; otherwise, the value shall be False.

1422 **7.3.11.3.5 Property: Query**

1423 The value of the Query property shall be a properly formed query statement that is conformant to the
 1424 requirements of the query language identified by the value of the QueryLanguage property, and that
 1425 states the coverage of the indication filter.

1426 **7.3.11.3.6 Property: QueryLanguage**

1427 The value of the QueryLanguage property shall identify the query language in which the query statement
 1428 exposed by the value of the Query property is expressed.

1429 NOTE This profile presently does not define a straight forward mechanism enabling clients to discover the set of
 1430 query languages supported by an implementation. A future version of this profile is expected to introduce
 1431 such a mechanism. For now, a rudimentary workaround may be inspecting the CIM representation of
 1432 existing indication filters, thereby discovery a lower boundary for the set of supported query languages.

1433 **7.3.11.4 Instance requirements**

1434 Indication filters (see 6.2) shall be represented by IndicationFilter instances in the Interop namespace.

1435 The representation in namespaces other than the Interop namespace should be avoided. However, if
 1436 additional IndicationFilter instances represent an indication filter also in implementation namespaces,
 1437 these instances shall have the same key property values as the one in the Interop namespace.

1438 **7.3.12 StaticIndicationFilter: CIM_IndicationFilter**

1439 **7.3.12.1 General**

1440 The requirements in this subclause are referencing profile and WBEM server related implementation
 1441 requirements.

1442 The StaticIndicationFilter adaptation models static indication filters; static indication filters are described in
 1443 6.2.3.

1444 The implementation type of the StaticIndicationFilter adaptation is: "abstract".

1445 **7.3.12.2 Element requirements**

1446 **7.3.12.2.1 General**

1447 Table 15 lists the element requirements for the StaticIndicationFilter adaptation.

1448 **Table 15 – StaticIndicationFilter: Element requirements**

Elements	Requirement	Description
Base adaptations		
IndicationFilter	Mandatory	See 7.3.11.
Properties		
QueryLanguage	Mandatory	See 7.3.12.2.2.
Operations		
CreateInstance()	Prohibited	The implementation shall return the CIM status code CIM_ERR_NOT_IMPLEMENTED.
DeleteInstance()	Prohibited	The implementation shall return the CIM status code CIM_ERR_NOT_IMPLEMENTED.
ModifyInstance()	Prohibited	The implementation shall return the CIM status code CIM_ERR_NOT_IMPLEMENTED.

1449 **7.3.12.2.2 Property: QueryLanguage**

1450 In adaptations based on the StaticIndicationFilter adaptation in referencing profiles owned by DMTF, the
 1451 value shall be "DMTF:CQL", thereby requiring CQL as the query language.

1452 **7.3.13 DynamicIndicationFilter: CIM_IndicationFilter**

1453 **7.3.13.1 General**

1454 The requirements in this subclause are WBEM server related implementation requirements.

1455 The DynamicIndicationFilter adaptation models dynamic indication filters; dynamic indication filters are
1456 described in 6.2.6.

1457 The requirement level of the DynamicIndicationFilter adaptation is conditional.

1458 Condition: The DynamicIndicationFilters feature is implemented; see 7.2.1.

1459 The implementation type of the DynamicIndicationFilter adaptation is: "instantiated".

1460 **7.3.13.2 Element requirements**

1461 **7.3.13.2.1 General**

1462 Table 16 lists the element requirements for the DynamicIndicationFilter adaptation.

1463 **Table 16 – DynamicIndicationFilter: Element requirements**

Elements	Requirement	Description
Base adaptations		
IndicationFilter	Mandatory	See 7.3.11.
Operations		
CreateInstance()	Mandatory	See 7.3.13.2.2.
DeleteInstance()	Mandatory	See 7.3.13.2.3.
ModifyInstance()	Optional	See 7.3.13.2.4.

1464 **7.3.13.2.2 Operation: CreateInstance()**

1465 Table 17 lists the error reporting requirements for the CreateInstance() operation on
1466 DynamicIndicationFilter instances. If any of the error situations described in the Description column of
1467 Table 17 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
1468 addition, the error reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

1469 **Table 17 – CreateInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the query language requested by the value of the Name property, as described in 7.3.11.3.2.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the query language requested by the value of the SourceNamespaces[] array property, as described in 7.3.11.3.3. Note that the identified local namespaces do not have to exist.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the query language requested by the value of the QueryLanguage property, as described in 7.3.11.3.6.

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Query property in the embedded CIM_IndicationFilter instance is not a well formed query statement in the implemented subset of the query language expressed by the value of the QueryLanguage property.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Query property in the embedded CIM_IndicationFilter instance covers lifecycle indications, but does not contain a WHERE clause.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the behavior requested by the value of the Query property, as described in 7.3.11.3.5.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the behavior requested by the value of the IndividualSubscriptionSupported property, as described in 7.3.11.3.4.
CIM_ERR_FAILED	Mandatory	The implementation is unable to create the requested dynamic indication filter for other unspecified reasons.

1470 If the CreateInstance() operation is successful, the requested dynamic indication filter shall be created,
 1471 and — as a consequence — shall be represented by a DynamicIndicationFilter instance in the requested
 1472 namespace.

1473 Clients should abstain from requesting the creation of DynamicIndicationFilter instances in namespaces
 1474 other than the Interop namespace. However, if the requested namespace is not the Interop namespace,
 1475 the implementation shall expose an additional DynamicIndicationFilter instance representing the dynamic
 1476 indication filter in the Interop namespace. That instance shall have identical values for all properties
 1477 except for the SourceNamespaces[] array property for which the provisions of 7.3.11.3.3 apply.

1478 If the CreateInstance() operation is fails, no dynamic indication filter shall be created, and — as a
 1479 consequence — no representing DynamicIndicationFilter instances shall be exposed in any namespace.

1480 **DEPRECATED**

1481 If the returned CIM status code is CIM_ERR_FAILED because an indication filter with the same coverage
 1482 as that requested already exists, the object path of the CIM_IndicationFilter instance representing the
 1483 existing indication filter in the Interop namespace shall be returned as the value of the ErrorSource
 1484 property in the CIM_Error instance accompanying the CIM status code.

1485 NOTE Only this specific ad-hoc use of CIM_Error is deprecated. It is intended that a future version of this profile
 1486 introduces extended error handling based on standard error messages.

1487 **DEPRECATED**

1488 With respect to input values for key properties the rules defined in [DSP1001](#) apply, namely that
 1489 implementation may ignore any input value for non-reference key properties, and that clients should
 1490 abstain from providing input values for key properties.

1491 **7.3.13.2.3 Operation: DeleteInstance()**

1492 Table 18 lists the error reporting requirements for the DeleteInstance() operation on
 1493 DynamicIndicationFilter instances, and related CIM status codes. If any of the error situations described
 1494 in the Description column of Table 18 matches, the operation shall fail and the corresponding CIM status
 1495 code shall be returned. In addition, the error reporting requirements defined in [DSP0223](#) for the
 1496 DeleteInstance() operation apply.

1497

Table 18 – DeleteInstance(): Error reporting requirements

Reporting mechanism	Requirement level	Description
CIM_ERR_FAILED	Mandatory	The represented dynamic indication filter is referenced by subscription(s).

1498 If the DeleteInstance() operation succeeds, the represented dynamic indication filter shall be deleted and
 1499 — as a consequence — no longer be represented by any DynamicIndicationFilter instances in any
 1500 namespace exposed by the implementation.

1501 NOTE The instance requirements of associations representing relationships of the deleted dynamic indication
 1502 filter imply that respective association instances in any namespace exposed by the implementation cease
 1503 to exist; in this case this applies to IndicationServiceOfIndicationFilter instances (see 7.3.14). However,
 1504 note that the DeleteInstance() operation for the dynamic indication filter is required to fail if subscriptions
 1505 exist.

1506 If the DeleteInstance() operation fails, the dynamic indication filter shall not be deleted, and — as a
 1507 consequence — any representing DynamicIndicationFilter instances shall continue to exist as before.

1508 **7.3.13.2.4 Operation: ModifyInstance()**

1509 The implementation of the ModifyInstance() operation enables clients to modify aspects of the behavior
 1510 of the represented indication filter.

1511 The requirement level of the ModifyInstance() operation is optional.

1512 Table 19 lists the error reporting requirements for the ModifyInstance() operation on
 1513 DynamicIndicationFilter instances. If any of the error situations described in the Description column of
 1514 Table 19 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1515 addition, the error reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1516

Table 19 – ModifyInstance(): Error reporting requirements

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the query language requested by the value of the Name property, as described in 7.3.11.3.2.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the query language requested by the value of the SourceNamespaces[] array property, as described in 7.3.11.3.3. Note that the identified local namespaces do not have to exist.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the query language requested by the value of the QueryLanguage property, as described in 7.3.11.3.6.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Query property in the embedded CIM_IndicationFilter instance is not a well formed query statement in the query language expressed by the value of the QueryLanguage property.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Query property in the embedded CIM_IndicationFilter instance covers lifecycle indications, but does not contain a WHERE clause.
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the behavior requested by the value of the Query property, as described in 7.3.11.3.5.

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the behavior requested by the value of the IndividualSubscriptionSupported property, as described in 7.3.11.3.4.
CIM_ERR_FAILED	Mandatory	The implementation is unable to apply the requested changes on the dynamic indication filter for other unspecified reasons.

1517 If the ModifyInstance() operation is successful, the requested modification on the dynamic indication filter
 1518 shall be applied, and — as a consequence — shall be reflected in all DynamicIndicationFilter instances
 1519 that represent the modified dynamic indication filter and are exposed by the implementation.

1520 If the ModifyInstance() operation is fails, the requested modification on the dynamic indication filter shall
 1521 not be applied, and — as a consequence — all DynamicIndicationFilter instances that represent the
 1522 dynamic indication filter shall remain unchanged.

1523 **7.3.13.3 Instance requirements**

1524 Dynamic indication filters shall be represented by DynamicIndicationFilter instances; the additional
 1525 requirements of 7.3.11.4 apply.

1526 **7.3.14 IndicationServiceOfIndicationFilter: CIM_ServiceAffectsElement**

1527 The requirements in this subclause are referencing profile and WBEM server related implementation
 1528 requirements.

1529 The IndicationServiceOfIndicationFilter adaptation models the relationship between indication services
 1530 and the indication filters they manage.

1531 The implementation type of the IndicationServiceOfIndicationFilter association adaptation is:
 1532 "instantiated".

1533 Table 20 lists the element requirements for the IndicationServiceOfIndicationFilter association adaptation.

1534 **Table 20 – IndicationServiceOfIndicationFilter: Element requirements**

Elements	Requirement	Description
Properties		
AffectingElement	Mandatory	Key: Value shall reference the IndicationService instance Multiplicity: 1
AffectedElement	Mandatory	Key: Value shall reference an IndicationFilter instance Multiplicity: *
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1535 Each IndicationService instance (see 7.3.2) shall be associated through an
 1536 IndicationServiceOfIndicationFilter instance with each IndicationFilter instance (see 7.3.11) representing
 1537 an indication filter managed by the indication service represented by the IndicationService instance.

1538 **7.3.15 IndicationSpecificIndicationFilter: CIM_IndicationFilter**

1539 **7.3.15.1 General**

1540 The requirements in this subclause are referencing profile and WBEM server related implementation
 1541 requirements.

1542 The IndicationSpecificIndicationFilter adaptation models indication-specific indication filters for indications
 1543 defined in referencing profiles or in this profile; indication-specific indication filters are described in 6.2.4.

1544 The requirement level of the IndicationSpecificIndicationFilter adaptation is optional.

1545 The IndicationSpecificIndicationFilter adaptation should be implemented if indications defined in a
 1546 referencing profile or in this profile are implemented.

1547 The implementation type of the IndicationSpecificIndicationFilter adaptation is: "instantiated".

1548 **7.3.15.2 Element requirements**

1549 **7.3.15.2.1 General**

1550 Table 21 lists the element requirements for the IndicationSpecificIndicationFilter adaptation.

1551 **Table 21 – IndicationSpecificIndicationFilter: Element requirements**

Element	Requirement	Description
Base adaptations		
StaticIndicationFilter	Mandatory	See 7.3.12.
Properties		
Name	Mandatory	See 7.3.15.2.2.
Query	Mandatory	See 7.3.15.2.3.

1552 **7.3.15.2.2 Property: Name**

1553 The value of the Name property shall be formatted as defined by the following ABNF rule:

```
1554   OrgID ":" RegisteredName ":" IndicationAdaptationName "Filter" [ "/"
1555   MessageIdentification ]
```

1556 OrgID and RegisteredName shall be specified as detailed in 7.3.11.3.2.

1557 IndicationAdaptationName shall be the name of the indication adaptation defined in the profile
 1558 identified by the RegisteredName rule. If the indication adaptation defines more than one possible
 1559 indication.

1560 The MessageIdentification suffix only applies for the representation of indication-specific indication
 1561 filters covering alert indications modeled by an adaptation based on the AlertIndication adaptation (see
 1562 7.3.31); in this case for each alert indication defined by an alert message reference in the profile, a
 1563 specific IndicationSpecificIndicationFilter instance is defined, where MessageIdentification shall be
 1564 set as defined in 7.3.31.2 for the CIM representation of the alert indication. Thus, for alert indications,

1565 there is a one-to-one relationship between defined referenced alert messages and possible
1566 corresponding IndicationSpecificIndicationFilter instances.

1567 For lifecycle indications the suffix is not necessary because adaptations based on the LifecycleIndication
1568 adaptation (see 7.3.32) only can address one event, as defined by a (constant) query statement. Thus,
1569 for lifecycle indications, there is a one-to-one relationship between defined lifecycle indications and
1570 possible corresponding IndicationSpecificIndicationFilter instances.

1571 7.3.15.2.3 Property: Query

1572 The value of the Query property shall be identical with the event definition query statement (see 7.3.29.2)
1573 of the indication adaptation defined in the referencing profile or in this profile that is covered by the
1574 represented indication-specific indication filter. In the case IndicationSpecificIndicationFilter instances
1575 covering alert indications modeled by an adaptation based on the AlertIndication adaptation, the value of
1576 the Query property shall apply the ABNF rule named `EventQuerySingle` (see 7.3.31.2); that way for
1577 alert indication adaptation referencing more than one alert message, separate
1578 IndicationSpecificIndicationFilter instances are defined for each referenced alert message.

1579 7.3.15.3 Instance requirements

1580 If a profile defines an indication adaptation based on the AlertIndication adaptation (see 7.3.31) or the
1581 Lifecycle adaptation (see 7.3.32), a corresponding indication-specific indication filter may be represented
1582 by an IndicationSpecificIndicationFilter instance, with respective values of the Name and Query
1583 properties.

1584 NOTE As with any indication filter (see 6.2.1), the existence of an indication-specific indication filter and its
1585 representation by an IndicationSpecificIndicationFilter instance does not imply that the covered indication
1586 is actually implemented. Furthermore, in the case where multiple implementations of the referencing profile
1587 exist in a WBEM server, multiple IndicationSpecificIndicationFilter instances with identical values for Name
1588 and Query properties may result.

1589 This profile leaves the decision whether or not to represent indication-specific indication filters as
1590 IndicationSpecificIndicationFilter instances to the implementation; however, referencing profiles can
1591 define an adaptation based on IndicationSpecificIndicationFilter adaptation that state more strict instance
1592 requirements.

1593 In any case, if an implementation decides to represent indication-specific indication filters, these are to be
1594 represented as required by the IndicationSpecificIndicationFilter adaptation. In addition, the requirements
1595 of related adaptations such as the ProfileSpecificFilterCollection adaptation (see 7.3.21) or the
1596 IndicationFilterInFilterCollection associations adaptation (see 7.3.19) apply.

1597 7.3.16 GlobalIndicationFilter: CIM_IndicationFilter

1598 7.3.16.1 General

1599 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1600 The GlobalIndicationFilter adaptation models a global indication filters; global indication filters are
1601 described in 6.2.5.

1602 The implementation type of the GlobalIndicationFilter adaptation is: "instantiated".

1603 7.3.16.2 Element requirements

1604 Table 22 lists the element requirements for the GlobalIndicationFilter adaptation.

1605

Table 22 – GlobalIndicationFilter: Element requirements

Element	Requirement	Description
Base adaptations		
StaticIndicationFilter	Mandatory	See 7.3.12.

1606

7.3.16.3 Instance requirements

1607

7.3.16.3.1 Instance requirements related to alert indications

1608

Table 23 lists the property value requirements for GlobalIndicationFilter instances covering all alert indications.

1609

1610

Table 23 – GlobalIndicationFilter: Instance requirements for instances covering all alert indications

1611

Value of Name property	Value of Query property
"DMTF:Indications:GlobalAlertIndicationFilter"	"SELECT * FROM CIM_AlertIndication"

1612

If the implementation supports the delivery of alert indications, it shall expose a GlobalIndicationFilter instance in the Interop namespace that complies with the value constraints defined in Table 23.

1613

1614

7.3.16.3.2 Instance requirements related to lifecycle indications

1615

Table 24 lists the property value requirements for GlobalIndicationFilter instances covering all lifecycle indications of a particular subtype.

1616

1617

Table 24 – GlobalIndicationFilter: Instance requirements for instances covering all lifecycle indications

1618

Value of Name property	Value of Query property
"DMTF:Indications:GlobalInstCreationIndicationFilter"	"SELECT * FROM CIM_InstCreation"
"DMTF:Indications:GlobalInstDeletionIndicationFilter"	"SELECT * FROM CIM_InstDeletion"
"DMTF:Indications:GlobalInstModificationIndicationFilter"	"SELECT * FROM CIM_InstModification"

1619

If the implementation supports the delivery of lifecycle indications, it shall expose a GlobalIndicationFilter instance in the Interop namespace for each row listed in Table 24 that complies with the value constraints defined in that row.

1620

1621

1622

7.3.17 StaticFilterCollection: CIM_FilterCollection

1623

7.3.17.1 General

1624

The requirements in this subclause are referencing profile and WBEM server related implementation requirements.

1625

1626

The StaticFilterCollection adaptation models static filter collections; static filter collections are described in 6.3.

1627

1628

The implementation type of the StaticFilterCollection adaptation is: "abstract".

1629 7.3.17.2 Semantical requirements

1630 The coverage of a filter collection shall be the aggregated coverage of all the indication gates contained
1631 by the filter collection. This definition applies recursively to contained filter collections.

1632 NOTE Since filter collections aggregate the coverages of contained indication filters and contained other filter
1633 collections, and do not specify a filter query statement on their own, the defined coverage of a static filter
1634 collection is finally described by the set of query statements of its (directly or indirectly) aggregated
1635 indication filters.

1636 The implementation shall filter all indications generated by (indication-specific parts of) the
1637 implementation that are within the coverage of a filter collection.

1638 The implementation shall ignore any generated indication that is outside the coverage of the filter
1639 collection.

1640 If a particular indication is within the coverage of more than one indication gate contained by a filter
1641 collection, that indication shall pass the filter collection only once, and shall not be replicated for every
1642 matching contained indication gate.

1643 Indications that passed a filter collection need to be further processed; see the requirements on the
1644 IndicationFilterName property defined in 7.3.29.4.2, and the semantical requirements on listener
1645 destinations defined in 7.3.23.2, and on subscriptions defined in 7.3.25.2. If implemented, the
1646 requirements for reliable indications as defined in 7.3.30 and 7.4 may apply.

1647 These semantics apply regardless of whether all, some or no contained indication gates are represented
1648 as collection members in CIM. Thus clients and listeners need to be aware of the fact that the coverage of
1649 a static filter collection may be larger than that observable through inspection of CIM represented
1650 members of that static filter collection. In other words, indications could be delivered to subscribed
1651 listeners that are within the coverage of members of the static filter collection that are not currently
1652 represented in CIM; in the extreme case no members at all are CIM represented. On the other hand,
1653 even if the coverage of a static filter collection is not represented through CIM, clients may have a priori
1654 knowledge about the defined coverage of that static filter collection, for example by means of built-in
1655 program code or data; see 7.3.17.3.

1656 NOTE During runtime, the set of members of a static filter collection and the extent to which such members are
1657 represented in CIM may change. For example, consider the global filter collection with a defined coverage
1658 covering all alert indications defined in referencing profiles, as defined in 7.3.22.4.1. Its member set might
1659 grow or shrink over time as implementations of referencing profiles are installed in or removed from the
1660 implementation; however, the conceptual defined coverage of "all alert indications defined in referencing
1661 profile" remains constant.

1662 7.3.17.3 Requirements pertaining to the defined coverage

1663 For concrete adaptations based (directly or indirectly) on the StaticFilterCollection adaptation, profiles
1664 shall specify a defined coverage (see 6.3.3.3) through normative text that identifies indication filters
1665 and/or other filter collections as the *contained members* of the static filter collection, and thereby —
1666 because of 7.3.17.2 — as contributors to the coverage of the static filter collection.

1667 NOTE If in a chain of (abstract and concrete) adaptations based on the StaticFilterCollection adaptation the
1668 defined coverage is defined as part of an intermediate (abstract or concrete) adaptation, that definition
1669 propagates into adaptations (directly or indirectly) based on that intermediate adaptation.

1670 The defined coverage of a static filter collection always applies regardless of whether any members are
1671 represented in CIM. For contained static filter collections the specification of a defined coverage is
1672 likewise required.

1673 The definition of the defined coverage may be specified at the level of adaptations, or may be broken
1674 down to individual adaptation instances, or both.

1675 For examples of how to specify a defined coverage, see 7.3.21.3 and 7.3.22.

1676 **7.3.17.4 Element requirements**

1677 **7.3.17.4.1 General**

1678 Table 25 lists the element requirements for the StaticFilterCollection adaptation.

1679 **Table 25 – StaticFilterCollection: Element requirements**

Element	Requirement	Description
Properties		
InstanceID	Mandatory	Key: See CIM schema definition.
CollectionName	Mandatory	See 7.3.17.4.2.
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .

1680 **7.3.17.4.2 Property: CollectionName**

1681 The value of the CollectionName property shall be formatted as defined by the following ABNF rule:

1682 `OrgID ":" RegisteredName ":" UniqueID`

1683 OrgID shall identify the business entity owning the referencing profile. OrgID shall include a copyrighted,
 1684 trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID
 1685 assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness,
 1686 OrgID shall not contain a colon (:).

1687 For referencing profiles owned by DMTF, OrgID shall match "DMTF".

1688 RegisteredName shall be the registered name of the referencing profile, as defined by the value of the
 1689 RegisteredName property in the RegisteredProfile instance representing the implemented version of the
 1690 referencing profile.

1691 UniqueID shall uniquely identify the instance within the implementation of the referencing profile.

1692 **DEPRECATED**

1693 For compatibility with version 1.0 of this profile, referencing profiles owned by business entities other than
 1694 DMTF may in addition define values for the Name property that are formatted as defined by the following
 1695 ABNF rule:

1696 `OrgID ":" UniqueID`

1697 Where:

- 1698 OrgID is defined above in this subclause.
- 1699 UniqueID shall uniquely identify the instance within the business entity owning the referencing
- 1700 profile.
- 1701 Version 1.1 of this profile has deprecated this additional format.

1702 **DEPRECATED**

1703 **7.3.17.5 Instance requirements**

1704 Static filter collections (see 6.3.3) shall be represented by StaticFilterCollection instances in the Interop

1705 namespace.

1706 The representation in namespaces other than the Interop namespace should be avoided. However, if

1707 additional StaticFilterCollection instances represent a static filter collection in implementation

1708 namespaces, these StaticFilterCollection instances shall have the same key property values as the one in

1709 the Interop namespace.

1710 If the FilterCollectionCoverageExposure feature (see 7.2.8) is available for a particular

1711 StaticFilterCollection instance, the contained members of the represented static filter collection (see

1712 7.3.17.3), and their containment relationship to the static filter collection are required to be represented in

1713 CIM; see 7.3.12 for the representation of contained static indication filters, see 7.3.17 for the

1714 representation of contained static filter collections, and see 7.3.19 and 7.3.20 for the representation of the

1715 containment relationship.

1716 **7.3.18 IndicationServiceOfFilterCollection: CIM_OwningCollectionElement**

1717 The requirements in this subclause are referencing profile and WBEM server related implementation

1718 requirements.

1719 The IndicationServiceOfFilterCollection adaptation models the relationship between a filter collection and

1720 the indication service that owns the filter collection.

1721 The implementation type of the IndicationServiceOfFilterCollection association adaptation is:

1722 "instantiated".

1723 Table 26 lists the element requirements for the IndicationServiceOfFilterCollection adaptation.

1724 **Table 26 – IndicationServiceOfFilterCollection: Element requirements**

Elements	Requirement	Description
Properties		
OwningElement	Mandatory	Key: Value shall reference the IndicationService instance Multiplicity: 1
OwnedElement	Mandatory	Key: Value shall reference the StaticFilterCollection instance Multiplicity: *
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1725 Each IndicationService instance (see 7.3.2.4) shall be associated through an
 1726 IndicationServiceOfFilterCollection instance to every StaticFilterCollection instance (see 7.3.17)
 1727 representing a static filter collection managed by the indication service represented by the
 1728 IndicationService instance.

1729 **7.3.19 IndicationFilterInFilterCollection: CIM_MemberOfCollection**

1730 The IndicationFilterInFilterCollection adaptation models the relationship between a filter collection and its
 1731 contained indication filters.

1732 The requirement level of the IndicationFilterInFilterCollection adaptation is conditional.

1733 Condition: The FilterCollectionCoverageExposure feature (see 7.2.8) is implemented.

1734 The implementation type of the IndicationFilterInFilterCollection association adaptation is: "instantiated".

1735 Table 27 lists the element requirements for the IndicationFilterInFilterCollection adaptation.

1736 **Table 27 – IndicationFilterInFilterCollection: Element requirements**

Elements	Requirement	Description
Properties		
Collection	Mandatory	Key: Value shall reference a StaticFilterCollection instance representing a filter collection containing indication filters Multiplicity: *
Member	Mandatory	Key: Value shall reference an StaticIndicationFilter instance representing a contained static indication filter Multiplicity: *
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1737 Each StaticFilterCollection (see 7.3.17) instance shall be associated through an
 1738 IndicationFilterInFilterCollection instance with each of the IndicationFilter (see 7.3.11) instances
 1739 representing contained indication filters.

1740 **7.3.20 FilterCollectionInFilterCollection: CIM_MemberOfCollection**

1741 The requirements in this subclause are referencing profile and WBEM server related implementation
 1742 requirements.

1743 The FilterCollectionInFilterCollection adaptation models the relationship between a filter collection and its
 1744 contained other filter collections.

1745 The requirement level of the FilterCollectionInFilterCollection adaptation is conditional.

1746 Condition: All of the following:

- 1747 • The static filter collections in the managed environment are capable of containing other static
 1748 filter collections
- 1749 • The FilterCollectionCoverageExposure feature (see 7.2.8) is implemented.

1750 The implementation type of the FilterCollectionInFilterCollection association adaptation is: "instantiated".

1751 Table 28 lists the element requirements for the FilterCollectionInFilterCollection adaptation.

1752 **Table 28 – FilterCollectionInFilterCollection: Element requirements**

Elements	Requirement	Description
Properties		
Collection	Mandatory	Key: Value shall reference a StaticFilterCollection instance representing a filter collection containing other filter collections Multiplicity: *
Member	Mandatory	Key: Value shall reference a StaticFilterCollection instance representing a contained filter collection Multiplicity: *
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

1753 Each StaticFilterCollection instance (see 7.3.17) representing a static filter collection that contains other
 1754 static filter collections shall be associated through a FilterCollectionInFilterCollection instance with each of
 1755 the StaticFilterCollection instances (see 7.3.17) representing a contained static filter collection.

1756 **7.3.21 ProfileSpecificFilterCollection: CIM_FilterCollection**

1757 **7.3.21.1 General**

1758 The requirements in this subclause are referencing profile and WBEM server related implementation
 1759 requirements.

1760 The ProfileSpecificFilterCollection adaptation models profile-specific filter collections; profile-specific filter
 1761 collections are described in 6.3.3.4.

1762 The requirement level of the ProfileSpecificFilterCollection adaptation is optional.

1763 The ProfileSpecificFilterCollection adaptation should be implemented.

1764 The implementation type of the ProfileSpecificFilterCollection adaptation is: "instantiated".

1765 **7.3.21.2 Element requirements**

1766 **7.3.21.2.1 General**

1767 Table 29 lists the element requirements for the ProfileSpecificFilterCollection adaptation.

1768 **Table 29 – ProfileSpecificFilterCollection: Element requirements**

Element	Requirement	Description
Base adaptations		
StaticFilterCollection	Mandatory	See 7.3.17.
Properties		

Element	Requirement	Description
CollectionName	Mandatory	See 7.3.21.2.2.

1769 **7.3.21.2.2 Property: CollectionName**

1770 The value of the CollectionName property shall be formatted as defined by the following ABNF rule:

```
1771     OrgID ":" RegisteredName ":"
1772     "ProfileSpecified" Type "IndicationFilterCollection"
```

1773 OrgID and RegisteredName shall be specified as detailed in 7.3.17.4.2.

1774 Type shall be "Alert" in case the represented profile-specific filter collection covers all alert indications,
 1775 and shall be "Lifecycle" in case the represented profile-specific filter collection covers all lifecycle
 1776 indications defined in the referencing profile identified by RegisteredName.

1777 NOTE This requirement does not preclude more than one instance in the Interop namespace from having
 1778 identical values for the CollectionName property, because, for example, the referencing profile could be
 1779 implemented more than once.

1780 **7.3.21.3 Requirements pertaining to the defined coverage**

1781 Requirements pertaining to the defined coverage are specified on a per instance basis; see 7.3.21.4
 1782 and 7.3.21.4.2.

1783 **7.3.21.4 Instance requirements**

1784 **7.3.21.4.1 Instance requirements for profile-specific filter collections covering all alert indications**
 1785 **specified in a profile**

1786 If and only if a referencing profile defines alert indications, the implementation may expose a
 1787 ProfileSpecificFilterCollection instance in the Interop namespace that covers all alert indications defined
 1788 in that profile. The element requirements defined in 7.3.21.2 apply.

1789 NOTE The existence of that ProfileSpecificFilterCollection instance does not imply that any alert indications are
 1790 actually implemented. Furthermore, in the case where multiple implementations of the referencing profile
 1791 exist in a WBEM server, multiple ProfileSpecificFilterCollection instances may result.

1792 The members of a profile-specific filter collection covering all alert indications defined in a referencing
 1793 profile shall be all indication-specific indication filters covering the alert indications defined in that
 1794 referencing profile; see 7.3.15. This definition in effect defines the defined coverage as all alert indications
 1795 defined in the referencing profile.

1796 NOTE For existing ProfileSpecificFilterCollection instances the instance requirements of association instances
 1797 representing relationships of the represented profile-specific filter collection apply; for example, see 7.3.18,
 1798 7.3.19 or 7.3.20.

1799 **7.3.21.4.2 Instance requirements for profile-specific filter collections covering all lifecycle**
 1800 **indications specified in a profile**

1801 If and only if a referencing profile defines lifecycle indications, the implementation may expose a
 1802 ProfileSpecificFilterCollection instance in the Interop namespace that covers all lifecycle indications
 1803 defined in that profile. The element requirements defined in 7.3.21.2 apply.

1804 NOTE The existence of such a ProfileSpecificFilterCollection instance does not imply that any lifecycle indications
 1805 are actually implemented. Furthermore, in the case where multiple implementations of the referencing
 1806 profile exist in a WBEM server, multiple ProfileSpecificFilterCollection instances may result.

1807 The members of a profile-specific filter collection covering all lifecycle indications defined in a referencing
 1808 profile shall be all indication-specific indication filters covering the lifecycle indications defined in that
 1809 referencing profile or in this profile; see 7.3.15. This definition in effect defines the defined coverage as all
 1810 lifecycle indications defined in the referencing profile.

1811 NOTE For existing ProfileSpecificFilterCollection instances the instance requirements of association instances
 1812 representing relationships of the represented profile-specific filter collection apply; for example, see 7.3.18,
 1813 7.3.19 or 7.3.20.

1814 The requirements specified in this subclause for lifecycle indications defined in referencing profiles shall
 1815 also apply for the lifecycle indications defined in this profile; see 7.3.33 and 7.3.34.

1816 **7.3.22 GlobalFilterCollection: CIM_FilterCollection**

1817 **7.3.22.1 General**

1818 The requirements in this subclause are referencing profile and WBEM server related implementation
 1819 requirements; see 7.1.

1820 The GlobalFilterCollection adaptation models global filter collection; global filter collections are described
 1821 in 6.3.3.5.

1822 The implementation type of the GlobalFilterCollection adaptation is: "instantiated".

1823 **7.3.22.2 Element requirements**

1824 Table 30 lists the element requirements for the GlobalFilterCollection adaptation.

1825 **Table 30 – GlobalFilterCollection: Element requirements**

Element	Requirement	Description
Base adaptations		
StaticFilterCollection	Mandatory	See 7.3.17.

1826 **7.3.22.3 Requirements pertaining to the defined coverage**

1827 Requirements pertaining to the defined coverage are specified on a per instance basis; see 7.3.22.4.1,
 1828 7.3.22.4.2, 7.3.22.4.3 and 7.3.22.4.4.

1829 **7.3.22.4 Instance requirements**

1830 **7.3.22.4.1 Instance requirements for the global filter collection covering all alert indications**
 1831 **specified in profiles**

1832 If any alert indications specified in referencing profiles or in this profile are implemented, the
 1833 implementation may expose a GlobalFilterCollection instance in the Interop namespace that covers all
 1834 alert indications defined in profiles. In implementations where it is not possible to determine whether alert
 1835 indications specified in referencing profiles are implemented, the instance may be exposed if the delivery
 1836 of alert indications is implemented in general.

1837 In the GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
 1838 following ABNF rule:

1839 "DMTF:Indications:"
 1840 "GlobalProfileSpecifiedAlertIndicationFilterCollection".

1841 In this case the members of the represented global filter collection shall be all profile-specific filter
 1842 collections covering the alert indications defined in any implemented referencing profile or in this profile;
 1843 see 7.3.21.4. This definition in effect specifies the defined coverage as all alert indications defined in
 1844 referencing profiles and in this profile; if instantiated by an implementation, the coverage would be all
 1845 implemented alert indications out of that set.

1846 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
 1847 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
 1848 or 7.3.20.

1849 **7.3.22.4.2 Instance requirements for the global filter collection covering all lifecycle indications** 1850 **specified in profiles**

1851 If any lifecycle indications specified in referencing profiles or in this profile are implemented, the
 1852 implementation may expose a GlobalFilterCollection instance in the Interop namespace that covers all
 1853 lifecycle indications defined in profiles. In implementations where it is not possible to determine whether
 1854 lifecycle indications specified in referencing profiles are implemented, the instance may be exposed if the
 1855 delivery of lifecycle indications is implemented in general.

1856 In GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
 1857 following ABNF rule:

```
1858     "DMTF:Indications:"  
1859     "GlobalProfileSpecifiedLifecycleIndicationFilterCollection".
```

1860 The members of the represented global filter collection shall be all profile-specific filter collections
 1861 covering the lifecycle indications defined in any implemented referencing profile or in this profile; see
 1862 7.3.21.4.2. This definition in effect specifies the defined coverage as all lifecycle indications defined in
 1863 referencing profiles and in this profile; if instantiated by an implementation, the coverage would be all
 1864 implemented lifecycle indications out of that set.

1865 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
 1866 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
 1867 or 7.3.20.

1868 **7.3.22.4.3 Instance requirements for the global filter collection covering all indications specified** 1869 **in profiles**

1870 If any indications specified in referencing profiles or in this profile are implemented, the implementation
 1871 may expose a GlobalFilterCollection instance in the Interop namespace that covers all indications defined
 1872 in profiles. In implementations where it is not possible to determine whether indications specified in
 1873 referencing profiles are implemented, the instance may be exposed if the delivery of indications is
 1874 implemented in general.

1875 In the GlobalFilterCollection instance, the value of the CollectionName property shall be as defined by the
 1876 following ABNF rule:

```
1877     "DMTF:Indications:"  
1878     "GlobalProfileSpecifiedIndicationFilterCollection"
```

1879 The members of the represented global filter collection shall be the following global filter collections (if
 1880 existing):

- 1881 • the global filter collection covering all alert indications defined in any implemented referencing
 1882 profile, as required in 7.3.22.4.1
- 1883 • the global filter collection covering all lifecycle indications defined in any implemented
 1884 referencing profile, as required in 7.3.22.4.2

1885 This definition in effect specifies the defined coverage as all indications defined in referencing profiles and
1886 in this profile; if instantiated by an implementation, the coverage would be all implemented indications out
1887 of that set.

1888 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
1889 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
1890 or 7.3.20.

1891 **7.3.22.4.4 Instance requirements for the global filter collection covering all lifecycle indications**

1892 If the implementation supports the delivery of lifecycle indications, the implementation shall expose a
1893 GlobalFilterCollection instance in the Interop namespace that covers all lifecycle indications defined in
1894 profiles.

1895 In GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
1896 following ABNF rule:

1897 `"DMTF:Indications:GlobalLifecycleIndicationFilterCollection".`

1898 The members of the represented global filter collection shall be all profile-specific filter collections
1899 covering the global indication filters that each cover all indications of one of the three subtypes of lifecycle
1900 indications (CIM_InstCreation, CIM_InstDeletion and CIM_InstModification); see 7.3.16.3.2.

1901 This definition in effect specifies the defined coverage as all lifecycle indications defined in referencing
1902 profiles and in this profile.

1903 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
1904 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
1905 or 7.3.20.

1906 **7.3.23 ListenerDestination: CIM_ListenerDestination**

1907 **7.3.23.1 General**

1908 The ListenerDestination adaptation models listener destinations; listener destinations are described in
1909 6.4.5.

1910 The implementation type of the ListenerDestination adaptation is: "instantiated".

1911 **7.3.23.2 Semantical requirements**

1912 For a particular listener destination, an implementation shall deliver any indication that passed the
1913 indication gate (see 6.2 or 6.3) referenced by any subscription (see 6.4.1) that also references the listener
1914 destination, to the listener referenced by that listener destination. See also the semantical requirements
1915 on indication filters defined in 7.3.11.2, on filter collections defined in 7.3.17.2, and on subscriptions
1916 defined in 7.3.25.2.

1917 NOTE It is possible that a particular indication is delivered more than once to a particular listener for various
1918 reasons, such as that the listener is referenced by more than one listener destination, or that the indication
1919 is within the coverage of more than one indication gate, each of which is referenced by a subscription
1920 referencing the listener destination referencing the listener.

1921 **7.3.23.3 Element requirements**

1922 **7.3.23.3.1 General**

1923 Table 31 lists the element requirements of the ListenerDestination adaptation.

1924

Table 31 – ListenerDestination Element requirements

Element	Requirement	Description
Properties		
Name	Mandatory	Key: See CIM schema definition.
CreationClassName	Mandatory	Key: See CIM schema definition.
SystemName	Mandatory	Key: See CIM schema definition.
SystemCreationClassName	Mandatory	Key: See CIM schema definition.
ElementName	Mandatory	See CIM schema description.
Destination	Mandatory	See 7.3.23.3.2.
PersistenceType	Mandatory	See 7.3.23.3.3.
Protocol	Mandatory	See CIM schema description.
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .
GetAssociatedInstancesWithPath()	Mandatory	See DSP0223 .
GetAssociatedInstancePaths()	Mandatory	See DSP0223 .
GetReferencingInstancesWithPath()	Mandatory	See DSP0223 .
GetReferencingInstancePaths()	Mandatory	See DSP0223 .
CreateInstance()	Optional	See 7.3.23.3.4 and DSP0223 .
DeleteInstance()	Optional	See 7.3.23.3.5 and DSP0223 .
ModifyInstance()	Optional	See 7.3.23.3.6 and DSP0223 .

1925 **7.3.23.3.2 Property: Destination**

1926 The value of the Destination property shall identify the listener referenced by the listener destination.

1927 A value of Null for the Destination property indicates a free listener destination (see 6.4.5).

1928 If the value of the Destination property is not Null, it shall be a valid IETF Uniform Resource Identifier
 1929 value (as defined in [RFC3986](#)) including the scheme, host and port as part of the URI Location.

1930 **7.3.23.3.3 Property: PersistenceType**

1931 The value of the PersistenceType property shall describe the durability of the represented listener
 1932 destination.

1933 The property values shall be constrained to 3 (Transient), 2 (Permanent), and Null.

1934 If the listener destination is permanent, then the value of the PersistenceType property shall be either Null
 1935 or 2 (Permanent). Permanent listener destinations are long-lived and are expected to be available for
 1936 indication delivery. For example, a typical listener referenced by a permanent listener destination would
 1937 be a system log file. The inability of an implementation to deliver indications to a listener referenced by a
 1938 permanent listener destination will be treated as an error condition by the implementation, as defined in
 1939 7.4.3.5.

1940 If the listener destination is transient, then the value of the PersistenceType property shall be 3
 1941 (Transient). Transient listener destinations are short-lived and have less strong requirements (than
 1942 permanent listener destinations) regarding their availability for indication delivery. For example, a typical
 1943 listener referenced by a transient listener destination would be a task progress meter in a graphical

1944 management application. The inability of an implementation to deliver indications to a listener described
 1945 by a transient listener destination will be handled by removing the listener destination and its
 1946 subscriptions from the implementation, as defined in 7.4.3.6.

1947 **7.3.23.3.4 Operation: CreateInstance()**

1948 Table 32 lists the error reporting requirements for the CreateInstance() operation on ListenerDestination
 1949 instances. If any of the error situations described in the Description column of Table 32 matches, the
 1950 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1951 reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

1952 **Table 32 – CreateInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The behavior requested by the value of the PersistenceType/OtherPersistenceType properties in the embedded CIM_ListenerDestination instance request a persistence type that is not implemented by the implementation.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Destination property in the embedded CIM_ListenerDestination instance does not constitute a valid URI as required in 7.3.23.3.2.
CIM_ERR_INVALID_PARAMETER	Mandatory	The behavior requested by the value of the Protocol/OtherProtocol properties in the embedded CIM_ListenerDestination instance request a protocol that is not implemented by the implementation.
CIM_ERR_FAILED	Mandatory	The number of listener destinations managed by the implementation would exceed the maximum number of listener destinations supported by the implementation; also see the description of the MaxListenerDestination property in 7.3.7.

1953 If the CreateInstance() operation is successful, the requested listener destination shall be created, and —
 1954 as a consequence — shall be represented by a ListenerDestination instance in the requested
 1955 namespace. In addition, if the requested namespace is not the Interop namespace, the implementation
 1956 shall expose an additional ListenerDestination instance representing the listener destination in the Interop
 1957 namespace (see 7.3.23.4).

1958 If the CreateInstance() operation fails, no listener destination shall be created, and — as a consequence
 1959 — no representing ListenerDestination instances shall be exposed in any namespace.

1960 The implementation may ignore the values of key properties in the embedded CIM_ListenerDestination
 1961 instance passed as the value of the NewInstance parameter.

1962 Clients should abstain from providing the values of key properties in the embedded
 1963 CIM_ListenerDestination instance passed as the value of the NewInstance parameter.

1964 Clients should abstain from requesting the creation of ListenerDestination instances in namespaces other
 1965 than the Interop namespace.

1966 Clients should favor the re-use of an existing listener destination referencing a particular listener over the
 1967 creation of a new listener destination referencing the same listener.

1968 **7.3.23.3.5 Operation: DeleteInstance()**

1969 Table 33 lists the error reporting requirements for the DeleteInstance() operation on ListenerDestination
 1970 instances, and related CIM status codes. If any of the error situations described in the Description column

1971 of Table 33 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1972 addition, the error reporting requirements defined in [DSP0223](#) for the DeleteInstance() operation apply.

1973 **Table 33 – ListenerDestination.DeleteInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_FAILED	Mandatory	The represented listener destination is referenced by subscription(s).

1974 If the DeleteInstance() operation is successful, the represented listener destination shall be deleted and
 1975 — as a consequence — shall no longer be represented by ListenerDestination instances in any
 1976 namespace exposed by the implementation.

1977 NOTE The instance requirements of associations representing relationships of the deleted listener destination
 1978 imply that respective association instances in any namespace exposed by the implementation cease to
 1979 exist; in this case this applies to IndicationServiceOfListenerDestination instances (see 7.3.24). However,
 1980 note that the DeleteInstance() operation for the listener destination is required to fail if subscriptions exist.

1981 If the DeleteInstance() operations fails, the listener destination shall not be deleted, and — as a
 1982 consequence — any representing ListenerDestination instances shall continue to exist as before.

1983 **7.3.23.3.6 Operation: ModifyInstance()**

1984 The ModifyInstance operation may be available for an instance of CIM_ListenerDestination.

1985 The implementation of the ModifyInstance() operation enables clients to modify existing listener
 1986 destinations.

1987 The requirement level of the ModifyInstance() operation is optional.

1988 Table 34 lists the error reporting requirements for the ModifyInstance() operation on ListenerDestination
 1989 instances. If any of the error situations described in the Description column of Table 34 matches, the
 1990 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1991 reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1992 **Table 34 – ModifyInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The behavior requested by the value of the PersistenceType/OtherPersistenceType properties in the embedded CIM_ListenerDestination instance request a persistence type that is not implemented by the implementation.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Destination property in the embedded CIM_ListenerDestination instance does not constitute a valid URI as required in 7.3.23.3.2.
CIM_ERR_INVALID_PARAMETER	Mandatory	The behavior requested by the value of the Protocol/OtherProtocol properties in the embedded CIM_ListenerDestination instance requests a protocol that is not implemented by the implementation.
CIM_ERR_FAILED	Mandatory	A modification of the Destination and/or the Protocol/OtherProtocol properties was requested, but the represented listener destination is still referenced by subscription(s).

1993 If the ModifyInstance() operation is successful, the requested modification on the listener destination
 1994 shall be applied, and — as a consequence — shall be reflected in all ListenerDestination instances that
 1995 represent the modified listener destination and are exposed by the implementation.

1996 If the ModifyInstance() operation fails, the requested modification on the listener destination shall not be
 1997 applied, and — as a consequence — all ListenerDestination instances that represent the listener
 1998 destination shall remain unchanged.

1999 **7.3.23.4 Instance requirements**

2000 Listener destinations (see 6.4.5) shall be represented by ListenerDestination instances in the Interop
 2001 namespace.

2002 The representation in namespaces other than the Interop namespace should be avoided. However, if
 2003 additional ListenerDestination instances represent the listener destination in implementation namespaces,
 2004 these ListenerDestination instances shall have the same key property values as the one in the Interop
 2005 namespace.

2006 **7.3.24 IndicationServiceOfListenerDestination: CIM_ServiceAffectsElement**

2007 The IndicationServiceOfListenerDestination adaptation models the relationship between indication
 2008 services and the listener destinations they manage. Indication services are described in 6.5.2; listener
 2009 destinations are described in 6.4.5.

2010 The implementation type of the IndicationServiceOfListenerDestination association adaptation is:
 2011 "instantiated".

2012 Table 35 lists the elements requirements of the IndicationServiceOfListenerDestination adaptation.

2013 **Table 35 – IndicationServiceOfListenerDestination: Element requirements**

Elements	Requirement	Description
Properties		
AffectingElement	Mandatory	Key: Value shall reference the IndicationService instance Multiplicity: 1
AffectedElement	Mandatory	Key: Value shall reference a ListenerDestination instance Multiplicity: *
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

2014 Each IndicationService (see 7.3.2) instance shall be associated through an
 2015 IndicationServiceOfListenerDestination instance with each ListenerDestination (see 7.3.23) instance
 2016 representing a listener destination managed by the indication service represented by the
 2017 IndicationService instance.

2018 **7.3.25 AbstractSubscription: CIM_AbstractIndicationSubscription**

2019 **7.3.25.1 General**

2020 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2021 The AbstractSubscription adaptation models subscriptions for the delivery of indications from an
 2022 indication gate to a listener referenced by a listener destination; subscriptions are described in 6.4.

2023 The implementation type of the AbstractSubscription association adaptation is: "abstract".

2024 **7.3.25.2 Semantical requirements**

2025 An implementation shall deliver any indication that passed the indication gate referenced by the
 2026 subscription (that is, any indication generated by the implementation that is within the coverage of the
 2027 indication gate) to the listener referenced by the listener destination referenced by the subscription.

2028 A listener that is referenced by the listener destination referenced by a subscription needs to be prepared
 2029 to receive any indication that is within the coverage of the indication gate referenced by that subscription.
 2030 Of course, listeners may ignore received indications.

2031 **7.3.25.3 Element requirements**

2032 Table 36 lists the element requirements for the AbstractSubscription adaptation.

2033 **Table 36 – AbstractSubscription: Element requirements**

Elements	Requirement	Description
Properties		
Filter	Mandatory	Key: Value shall reference the IndicationFilter instance or the StaticFilterCollection instance
Handler	Mandatory	Key: Value shall reference the ListenerDestination instance
OnFatalErrorPolicy	Mandatory	See 7.3.25.3.1.
OtherOnFatalErrorPolicy	Conditional	Condition: The OnFatalErrorPolicy property can have the value 1 (Other). Pattern (".-") Value shall be non-Null if the value of the OnFatalErrorPolicy property is 1 (Other).
FailureTriggerTimeInterval	Mandatory	Value shall be the minimum delay before the policy indicated by the value of the OnFatalErrorPolicy property is applied
SubscriptionState	Mandatory	See CIM schema definition.

Elements	Requirement	Description
OtherSubscriptionState	Conditional	Condition: The SubscriptionState property can have the value 1 (Other). Pattern (".+") Value shall be non-Null if the value of the SubscriptionState property is 1 (Other).
RepeatNotificationPolicy	Mandatory	See 7.3.25.3.2.
RepeatNotificationInterval	Conditional exclusive	See 7.3.25.3.3.
RepeatNotificationGap	Conditional exclusive	See 7.3.25.3.4.
RepeatNotificationCount	Conditional exclusive	See 7.3.25.3.5.
Operations		
DeleteInstance()	Mandatory	See 7.3.25.3.6 and DSP0223 .
ModifyInstance()	Optional	See 7.3.25.3.7 and DSP0223 .
NOTE The CreateInstance() operation is defined in adaptations based on the AbstractSubscription adaptation; see 7.3.26 and 7.3.27.		

2034 **7.3.25.3.1 Property: OnFatalErrorPolicy**

2035 The value of the OnFatalErrorPolicy property shall indicate the behavior that the implementation exposes
2036 with respect to represented subscriptions in case of failures that imply that some aspect of indication
2037 generation processing or indication delivery is no longer functioning and indications may be lost.

2038 A value of 4 (Remove) shall indicate that the implementation performs implicit subscription removal as
2039 detailed in 7.4.3.6; this shall be the default behavior.

2040 **7.3.25.3.2 Property: RepeatNotificationPolicy**

2041 The value of the RepeatNotificationPolicy property shall indicate the policy that the implementation
2042 applies with respect to the avoidance of repeated indication delivery of repeated indications as described
2043 in 6.1.6.

2044 Table 37 lists constraints for the value of the RepeatNotificationPolicy property.

2045 **Table 37 – RepeatNotificationPolicy: Value constraints**

Subscription behavior for the avoidance of repeated indication delivery	Required value
No avoidance of repeated indication delivery	2 (None)
The implementation applies the policy of suppressing the repeated indication delivery for the represented subscription, as described in 6.1.6.	3 (Suppress)
The implementation applies the policy of delaying the repeated indication delivery for the represented subscription, as described in 6.1.6 .	4 (Delay)

2046 **7.3.25.3.3 Property: RepeatNotificationInterval**

2047 The requirement level of the RepeatNotificationInterval property is conditional exclusive.

2048 Condition: Either the SuppressRepeatNotificationPolicy feature (see 7.2.5) or the
2049 DelayRepeatNotificationPolicy feature (see 7.2.6) is available.

2050 If the implementation applies the SuppressRepeatNotificationPolicy feature (see 7.2.5) for the
2051 represented subscription, as indicated by the value 3 (Suppress) for the RepeatNotification property, the
2052 value of the RepeatNotificationInterval property shall be the length of the time interval in seconds that the
2053 implementation waits after initial delivery of a number of repeated indications as indicated by the value of
2054 the RepeatNotificationCount property before delivering the next repeated indication.

2055 If the implementation applies the DelayRepeatNotificationPolicy feature (see 7.2.6) for the represented
2056 subscription, as indicated by the value 4 (Delay) for the RepeatNotification property, the value of the
2057 RepeatNotificationInterval property shall be the length of the monitoring time interval in seconds during
2058 which the implementation monitors the indication gate referenced by the subscription for a number of
2059 additional repeated indications. Furthermore, only if during that monitoring interval at least the number of
2060 repeated indications as indicated by the value of the RepeatNotificationCount accrue, delivers only the
2061 first indication as a substitute for all the repeated indications accrued during the monitoring time interval.

2062 **7.3.25.3.4 Property: RepeatNotificationGap**

2063 The requirement level of the RepeatNotificationGap property is conditional exclusive.

2064 Condition: The DelayRepeatNotificationPolicy feature (see 7.2.6) is implemented.

2065 The value of the RepeatNotificationGap property shall be the length of the delay time interval in seconds
2066 that the implementation waits after delivering the first of a number of repeated indications that accrued
2067 during the monitoring time interval, before starting another monitoring time interval, as described in
2068 7.3.25.3.5 with respect to implementations of the DelayRepeatNotificationPolicy feature.

2069 **7.3.25.3.5 Property: RepeatNotificationCount**

2070 The requirement level of the RepeatNotificationCount property is conditional exclusive.

2071 Condition: Either the SuppressRepeatNotificationPolicy feature (see 7.2.5) or the
2072 DelayRepeatNotificationPolicy feature (see 7.2.6) is implemented.

2073 If the implementation applies the SuppressRepeatNotificationPolicy feature (see 7.2.5) for the
2074 represented subscription, as indicated by the value 3 (Suppress) for the RepeatNotification property, the
2075 value of the RepeatNotificationCount property shall be the number of repeated indications that the
2076 implementation delivers before suppressing the delivery of further repeated indications within the time
2077 interval exposed by the value of the RepeatNotificationInterval property.

2078 If the implementation applies the DelayRepeatNotificationPolicy feature (see 7.2.6) for the represented
2079 subscription, as indicated by the value 4 (Delay) for the RepeatNotification property, the value of the
2080 RepeatNotificationCount property shall be the number of repeated indications that the implementation is
2081 required to monitor and delay during the monitoring time interval exposed by the value of the
2082 RepeatNotificationInterval property. Only if during that monitoring time interval the number of accrued
2083 repeated indications reaches that number, the implementation shall deliver the first of repeated indication
2084 as a substitute for the accrued repeated indications. In other words, the quotient of the values of the
2085 RepeatNotificationCount and the RepeatNotificationInterval properties expresses a rate of repeated
2086 indications that must have been reached or exceeded during the monitoring time interval before one
2087 indication is delivered at the end of the monitoring time interval.

2088 **7.3.25.3.6 Operation: DeleteInstance()**

2089 The error situations and CIM status codes defined in [DSP0223](#) for the DeleteInstance() operation apply.

2090 If the DeleteInstance() operation succeeds, the represented subscription shall be deleted and — as a
2091 consequence — shall no longer be represented by any AbstractSubscription instances in any namespace
2092 exposed by the implementation.

2093 If the DeleteInstance() operation fails, the subscription shall not be deleted, and — as a consequence —
2094 any representing AbstractSubscription instances shall continue to exist as before.

2095 **7.3.25.3.7 Operation: ModifyInstance()**

2096 The requirement level of the ModifyInstance() operation is optional.

2097 Table 38 lists the error reporting requirements for the ModifyInstance() operation on AbstractSubscription
 2098 instances, and related CIM status codes. If any of the error situations described in the Description column
 2099 of Table 38 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 2100 addition, the error reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation are
 2101 applicable.

2102 **Table 38 – ModifyInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the OnFatalErrorPolicy/OtherOnFatalErrorPolicy properties (see 7.3.25.3.1) in the embedded CIM_AbstractSubscription instance request a fatal error policy that is not supported by the implementation, or the implementation does not support client-initiated changes of the fatal error policy.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the FailureTriggerTimeInterval property in the embedded CIM_AbstractSubscription instance requests a time interval that is not supported by the implementation, or the implementation does not support client-initiated changes of the failure trigger time interval.
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the RepeatNotificationPolicy/RepeatNotificationInterval-/RepeatNotificationGap/RepeatNotificationCount properties in the embedded CIM_AbstractSubscription instance request a change in the repeat notification behavior of the represented subscription state that is not supported by the implementation, or the implementation does not support client-initiated changes of the repeat notification behavior.
CIM_ERR_INVALID_PARAMETER	Mandatory	The embedded CIM_AbstractSubscription instance has non-Null values for properties for which the implementation does not support client-initiated modifications.

2103 If the ModifyInstance() operation is successful, the requested modification on the represented
 2104 subscription shall be applied, and — as a consequence — shall be reflected in all AbstractSubscription
 2105 instances that represent the modified subscription.

2106 If the ModifyInstance() operation fails, the requested modification on the subscription shall not be
 2107 applied, and — as a consequence — all AbstractSubscription instances that represent the subscription
 2108 shall remain unchanged.

2109 **7.3.25.4 Instance requirements**

2110 Subscriptions (see 6.4.1) shall be represented by AbstractSubscription instances in the Interop
 2111 namespace that relate either IndicationFilter instances (see 7.3.11) or StaticFilterCollection instances
 2112 (see 7.3.17) with ListenerDestination instances (see 7.3.23).

2113 The representation in namespaces other than the Interop namespace should be avoided. However, if
 2114 both the indication filter/filter collection and the related listener destination represented by the referenced
 2115 instances in the Interop namespace are also represented by additional instances in other namespaces,
 2116 respective AbstractSubscription instances shall represent the subscription in these other namespaces as
 2117 well.

2118 **7.3.26 FilterSubscription: CIM_IndicationSubscription**

2119 **7.3.26.1 General**

2120 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2121 The FilterSubscription adaptation models subscriptions for the delivery of indications from an indication
 2122 filter to a listener referenced by a listener destination; subscriptions are described in 6.4.

2123 The requirement level of the FilterSubscription adaptation is conditional.

2124 Condition: The IndividualFilterSubscription feature (see 7.2.7) is implemented.

2125 The implementation type of the FilterSubscription association adaptation is: "instantiated".

2126 **7.3.26.2 Semantical requirements**

2127 The semantical requirements of 7.3.25.2 apply respectively for the FilterSubscription adaptation.

2128 **7.3.26.3 Element requirements**

2129 **7.3.26.3.1 General**

2130 Table 39 lists the element requirements for the FilterSubscription adaptation.

2131 **Table 39 – FilterSubscription: Element requirements**

Elements	Requirement	Description
Base adaptations		
AbstractSubscription	Mandatory	See 7.3.25.
Properties		
Filter	Mandatory	Key: Value shall reference the IndicationFilter instance Multiplicity: *
Handler	Mandatory	Key: Value shall reference the ListenerDestination instance Multiplicity: *
Operations		
CreateInstance()	Mandatory	See 7.3.26.3.2 and DSP0223 .

2132 **7.3.26.3.2 Operation: CreateInstance()**

2133 Table 40 lists the error reporting requirements for the CreateInstance() operation on FilterSubscription
 2134 instances. If any of the error situations described in the Description column of Table 40 matches, the
 2135 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 2136 reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

2137

Table 40 – CreateInstance(): Error reporting requirements

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Filter property in the embedded CIM_IndicationSubscription instance references an instance that does not exist, or is not an IndicationFilter instance (see 7.3.11).
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Handler property in the embedded CIM_IndicationSubscription instance references an instance that does not exist, or is not ListenerDestination instance (see 7.3.23).
CIM_ERR_FAILED	Mandatory	The IndividualFilterSubscription feature (see 7.2.7) is not available for the indication filter represented by the IndicationFilter instance referenced by the value of the IndicationFilter property in the embedded CIM_IndicationSubscription instance.
CIM_ERR_FAILED	Mandatory	The number of subscriptions managed by the implementation would exceed the maximum number of subscriptions supported by the implementation; also see the description of the MaxSubscriptions property in 7.3.7.
NOTE With version 1.2 of this profile the requirements for CIM status code values were refined, fixing the incorrect requirement for a value named CIM_ERROR_NOT_SUPPORTED mandated by previous versions.		

2138 If the CreateInstance() operation is successful, the requested filter subscription was created, and
 2139 consequently — as required by 7.3.26.4 — shall be represented by a FilterSubscription instance in the
 2140 requested namespace. In addition, if the requested namespace is not the Interop namespace, the
 2141 implementation shall expose an additional FilterSubscription instance representing the subscription in the
 2142 Interop namespace (see 7.3.26.4).

2143 If the CreateInstance() operation fails, no subscription shall be created, and — as a consequence — no
 2144 representing FilterSubscription instances shall be exposed in any namespace.

2145 Clients should abstain from requesting the creation of FilterSubscription instances in namespaces other
 2146 than the Interop namespace.

2147 **7.3.26.4 Instance requirements**

2148 The requirements of 7.3.25.4 apply respectively for FilterSubscription instances.

2149 **7.3.27 CollectionSubscription: CIM_FilterCollectionSubscription**

2150 **7.3.27.1 General**

2151 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2152 The CollectionSubscription adaptation models subscriptions for the delivery of indications from a filter
 2153 collection to a listener referenced by a listener destination; subscriptions are described in 6.4.

2154 The implementation type of the FilterCollectionSubscription association adaptation is: "instantiated".

2155 **7.3.27.2 Semantical requirements**

2156 The semantical requirements of 7.3.25.2 apply respectively for the CollectionSubscription adaptation.

2157 **7.3.27.3 Element requirements**

2158 **7.3.27.3.1 General**

2159 Table 41 lists the element requirements for the CollectionSubscription adaptation.

2160 **Table 41 – CollectionSubscription: Element requirements**

Elements	Requirement	Description
Base adaptations		
AbstractSubscription	Mandatory	See 7.3.25.
Properties		
Filter	Mandatory	Key: Value shall reference the StaticFilterCollection instance Multiplicity: *
Handler	Mandatory	Key: Value shall reference the ListenerDestination instance Multiplicity: *
Operations		
CreateInstance()	Mandatory	See 7.3.27.3.2 and DSP0223 .

2161 **7.3.27.3.2 Operation: CreateInstance()**

2162 Table 42 lists the error reporting requirements for the CreateInstance() operation on
 2163 CollectionSubscription instances. If any of the error situations described in the Description column of
 2164 Table 42 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 2165 addition, the error reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

2166 **Table 42 – CreateInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Collection property in the embedded CIM_FilterCollectionSubscription instance references an instance that does not exist, or is not a StaticFilterCollection instance (see 7.3.17).
CIM_ERR_INVALID_PARAMETER	Mandatory	The value of the Handler property in the embedded CIM_FilterCollectionSubscription instance references an instance that does not exist, or is not a ListenerDestination instance (see 7.3.23).
CIM_ERR_FAILED	Mandatory	The number of subscriptions managed by the implementation would exceed the maximum number of subscriptions supported by the implementation; also see the description of the MaxSubscriptions property in 7.3.7.
NOTE With version 1.2 of this profile the requirements for CIM status code values were refined, fixing the incorrect requirement for a value named CIM_ERROR_NOT_SUPPORTED mandated by previous versions.		

2167 If the CreateInstance() operations is successful, the requested filter subscription was created, and
 2168 consequently — as required by 7.3.27.4 — shall be represented by a CollectionSubscription instance in
 2169 the requested namespace. In addition, if the requested namespace is not the Interop namespace, the
 2170 implementation shall expose an additional CollectionSubscription instance representing the subscription
 2171 in the Interop namespace (see 7.3.27.4).

2172 If the CreateInstance() operation fails, no subscription shall be created, and — as a consequence — no
 2173 representing CollectionSubscription instances shall be exposed in any namespace.

2174 Clients should abstain from requesting the creation of CollectionSubscription instances in namespaces
 2175 other than the Interop namespace.

2176 **7.3.27.4 Instance requirements**

2177 The instance requirements of 7.3.25.4 apply respectively for CollectionSubscription instances.

2178 **DEPRECATED**

2179 **7.3.28 ProfileOfFilterCollection: CIM_ConcreteDependency**

2180 The ProfileOfFilterCollection adaptation models the relationship between a filter collection defined in a
 2181 referencing profile and the profile registration of that referencing profile.

2182 The implementation type of the ProfileOfFilterCollection association adaptation is: "instantiated".

2183 Each StaticFilterCollection instance (see 7.3.17) representing a filter collection defined in a referencing
 2184 profile shall be associated through a ProfileOfFilterCollection instance with the ProfileRegistration
 2185 instance (see [DSP1033](#)) representing the implemented version of the referencing profile.

2186 NOTE This profile assumes that a future version of the Profile Registration profile (see [DSP1033](#)) will be based
 2187 on version 1.1 of the Profile Usage Guide (see [DSP1001](#)), and define the ProfileRegistration adaptation;
 2188 until then, substitute that by the definition of the CIM_RegisteredProfile "profile class" defined in version
 2189 1.0 of [DSP1033](#).

2190 Table 43 lists the element requirements for the ProfileOfFilterCollection adaptation.

2191 **Table 43 – ProfileOfFilterCollection: Element requirements**

Elements	Requirement	Description
Properties		
Antecedent	Mandatory	Key: Value shall reference the ProfileRegistration instance Multiplicity: 1
Dependent	Mandatory	Key: Value shall reference the StaticFilterCollection instance Multiplicity: *
Operations		
GetInstance()	Mandatory	See DSP0223 .
GetClassInstancesWithPath()	Mandatory	See DSP0223 .
GetClassInstancePaths()	Mandatory	See DSP0223 .

2192 **DEPRECATED**

2193 **7.3.29 BasicIndication: CIM_Indication**

2194 **7.3.29.1 General**

2195 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2196 The BasicIndication adaptation models indications; indications are described in 6.1.

- 2197 The implementation type of the BasicIndication indication adaptation is: "abstract".
- 2198 **7.3.29.2 Event definition requirements**
- 2199 Referencing profiles that model indications through adaptations based on the BasicIndication adaptation
 2200 shall define event that the indication is designed to report. This event definition shall be accomplished by
 2201 means of an event definition query statement stated in CQL (see [DSP0202](#)).
- 2202 The purpose of an event definition query statement is to formally define the event(s) that an indication
 2203 adaptation is designed to report, such that by inspecting the event definition query statements an
 2204 implementer knows how to implement the indication adaptation. A CIM representation of event definition
 2205 query statements is not defined, thus there is no requirement for implementations or clients to be able to
 2206 programmatically interpret event definition query statements.
- 2207 **NOTE** Event definition query statements are different from indication filter query statements. An indication filter
 2208 query statement (see 7.3.11.3.5) defines the coverage of an indication filter, and is exposed to clients by
 2209 the value of the Query property in the IndicationFilter instance representing the indication filter. The
 2210 IndicationSpecificIndicationFilter adaptation (see 7.3.15) models indication-specific indication filters (see
 2211 6.2.4) and addresses the needs of clients requiring notifications about events reported by particular
 2212 indications specified in a profile.
- 2213 The CQL query statement defining the event shall comply with the following ABNF rule:
- 2214 `"SELECT" WS PropertySet WS "FROM" WS IndicationClass WS "WHERE" WS`
 2215 `SelectionExpression`
- 2216 `PropertySet` shall be `"*"`, or a comma-separated list of property names.
- 2217 `IndicationClass` shall be the adapted indication class, that is, `CIM_Indication` or a subclass thereof.
- 2218 `SelectionExpression` shall be a constant string that defines a selection expression conformant with
 2219 the rules for selection expressions defined by [DSP0202](#).
- 2220 `WS` represents one or more whitespace characters.
- 2221 The requirements in this subclause may be refined by requirements defined in adaptations based on the
 2222 BasicIndication adaptation, including the case that a refined query statement references an external
 2223 element (such as an alert message definition in a message registry) that defines the event.
- 2224 **7.3.29.3 Indication origin**
- 2225 Each indication shall be assigned an origin namespace (see 6.1.2.4).
- 2226 In general, an implementation is free to select any local namespace as the origin namespace for a
 2227 generated indication; however, adaptations based on the BasicIndication adaptation such as the
 2228 AlertIndication adaptation (see 7.3.31) and the LifecycleIndication (see 7.3.32) establish additional
 2229 constraints.
- 2230 The indication origin is not represented in the CIM representation of an indication as defined by the
 2231 `CIM_Indication` class.
- 2232 The implementation class of the indication is required to reside in the origin namespace.
- 2233 **NOTE** As with any implementation class, the existence of an indication implementation class within a namespace
 2234 is does not sufficiently indicate that the indication is really implemented. Additional requirements — such
 2235 as the presence and integration of functional code implementing the indication — apply, but are outside of
 2236 the scope of this profile.
- 2237 The indication origin is required to be considered during indication filtering; see 6.1.4 and 7.3.11.2.

2238 **7.3.29.4 Element requirements**

2239 **7.3.29.4.1 General**

2240 Table 44 lists the element requirements for the BasicIndication adaptation.

2241 **Table 44 – BasicIndication: Element requirements**

Elements	Requirement	Description
Properties		
IndicationFilterName	Mandatory	See 7.3.29.4.2.
IndicationIdentifier	Mandatory	See CIM schema definition.
IndicationTime	Mandatory	See CIM schema definition.

2242 **7.3.29.4.2 Property: IndicationFilterName**

2243 The value of the IndicationFilterName property shall contain the name of the indication gate that the
 2244 indication passed before being delivered to the listeners subscribed to that indication gate. For indication
 2245 filters, the name is exposed by the value of the Name property in representing IndicationFilter instances
 2246 (see 7.3.11). For filter collections, the name is exposed by the value of the CollectionName property in
 2247 representing StaticFilterCollection instances (see 7.3.17).

2248 Because an indication is generated independently and before it is subjected to filtering, the name of the
 2249 filtering indication gate is not known at indication-generation time. Instead, a generated indication might
 2250 match a large number of indication gates. During indication filtering (see 6.1.4 and 7.3.11.2), each time a
 2251 generated indication matches an indication gate with existing subscriptions, and before delivering that
 2252 indication to subscribed listeners, the implementation shall set the value of the IndicationFilterName
 2253 property in the BasicIndication instance representing the indication to the identification of that indication
 2254 gate, as follows:

- 2255 • in case of indication filters, the identification shall be the value of the Name property of the
 2256 IndicationFilter instance representing the indication filter
- 2257 • in case of filter collections, the identification shall be the value of the CollectionName property of
 2258 the StaticFilterCollection instance representing the filter collection.

2259 NOTE 1 The requirement for referencing filter collections was added with version 1.2. of this profile.

2260 NOTE 2 A listener may use the value of the IndicationFilterName property to determine which indication gate was
 2261 passed by the indication before being delivered to the listener.

2262 **7.3.29.5 Indication generation requirements**

2263 Adaptations based on the BasicIndication adaptation are required to define the event that the modeled
 2264 indication is designed to report; see 7.3.29.2.

2265 If the event defined by such an adaptation occurs, and if subscriptions exist for any indication gate
 2266 covering the modeled indication, an instance of the indication adaptation based on the BasicIndication
 2267 shall be generated.

2268 NOTE The way this requirement is stated it provides for the optimized approach of checking for the presence of
 2269 matching indication gate with subscriptions already at indication generation time; however, even in this
 2270 case indication filtering is required as a subsequent step (see 6.1.4) in order to ensure that all matching
 2271 indication gates are considered, and indication delivery occurs to all listeners subscribed to any of the
 2272 indication gates covering the indication.

2273 **7.3.30 ReliableIndication: CIM_Indication**

2274 **7.3.30.1 General**

2275 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2276 The ReliableIndication adaptation models reliable indications; the concept of reliable indications is
 2277 introduced in 6.1.5. Additional requirements for reliable indication delivery are specified in 7.4.

2278 The implementation type of the ReliableIndication indication adaptation is: "abstract".

2279 NOTE The ReliableIndications adaptation is intentionally not based on the BasicIndication adaptation, such that it
 2280 can be implemented independently as a separate option. Reliable indication delivery is typically
 2281 implemented centrally once for the delivery of all indications implemented by an implementation.

2282 **7.3.30.2 Element requirements**

2283 **7.3.30.2.1 General**

2284 Table 45 lists the element requirements for the ReliableIndication adaptation.

2285 **Table 45 – ReliableIndication: Element requirements**

Elements	Requirement	Description
Properties		
SequenceContext	Mandatory	See 7.3.30.2.2.
SequenceNumber	Mandatory	See 7.3.30.2.3.

2286 **7.3.30.2.2 Property: SequenceContext**

2287 The value of the SequenceContext property shall contain the sequence context portion of the sequence
 2288 identifier (see 3.30 and 7.4.2). See the CIM schema description for additional constraints and the required
 2289 semantics, and see 7.4 for additional requirements on reliable indication delivery.

2290 NOTE 1 The CIM schema definition of the CIM_Indication class requires for the SequenceContext property that the
 2291 implementation maintains the context for this property separately for each registered listener destination,
 2292 and that restarts of the WBEM server cause the value to change. This requirement enables a listener to
 2293 detect WBEM server restarts, and to differentiate the indication streams from a particular WBEM server
 2294 that were processed (within that WBEM server) through different listener destinations referring to the
 2295 listener.

2296 NOTE 2 Indications can be lost when a listener fails and restarts, with the WBEM server continuing to send
 2297 indications while the listener is inactive. In that case, upon restart of the listener, if does not persist the last
 2298 received sequence identifier, the listener would establish the sequence identifier of the first received
 2299 indication after the restart as check value, failing to notice that while it was inactive additional indications
 2300 were sent (and lost). One approach for discovering an actual loss of indications might be to persist the
 2301 latest sequence identifier as part of a listener termination routine, and upon restart use the persisted value
 2302 as a check value (instead of that taken from the first arriving indication after the restart).

2303 **7.3.30.2.3 Property: SequenceNumber**

2304 The value of the SequenceNumber property shall contain the sequence number portion of the sequence
 2305 identifier (see 3.30 and 7.4.2). See the CIM schema description for additional constraints and the required
 2306 semantics, and see 7.4 for additional requirements on reliable indication delivery.

2307 NOTE The CIM schema definition of CIM_Indication class requires for the SequenceNumber property in the
 2308 stream of instances processed through a particular listener destination, that the value starts at 0 whenever
 2309 the value of the SequenceContext property changes.

2310 **7.3.31 AlertIndication: CIM_AlertIndication**2311 **7.3.31.1 General**

2312 The AlertIndication adaptation models alert indications; alert indications are described in 6.1.3.

2313 The implementation type of the AlertIndication indication adaptation is: "abstract".

2314 It is expected that the AlertIndication adaptation is used as a base adaptation for modeling alert
2315 indications in referencing profiles.

2316 **7.3.31.2 Event definition requirements**

2317 This subclause refines the event definition requirements established by the BasicIndication adaptation;
2318 see 7.3.29.2.

2319 The query statement defined by the following ABNF rules define the event(s) that are reported by
2320 AlertIndication instances:

- 2321 • If the AlertIndication adaptation identifies only one related alert message (see 7.3.31.3), the
2322 event query statement is defined as follows:

```
2323 EventQuerySingle = "SELECT" WS PropertySet WS "FROM" WS
2324 AlertIndicationClass WS "WHERE" WS "OwningEntity='" OwningEntity "'"
2325 WS "AND" WS "MessageID=" MessageId WS AdditionalWhereElements
```

- 2326 • If the AlertIndication adaptation identifies more than one related alert message (see 7.3.31.3),
2327 the event query statement is defined as follows:

```
2328 EventQueryMulti = "SELECT" WS PropertySet WS "FROM" WS
2329 AlertIndicationClass WS "WHERE" WS "OwningEntity='" OwningEntity "'"
2330 WS "AND" WS "MessageID LIKE" WS "'" MessageSet "'" [ WS
2331 AdditionalSelectionExpression ]
```

```
2332 MessageSet = MessageIdentification [ "|" MessageSet ]
```

2333 NOTE Recall that the purpose of the event definition query statement is to formally define the event(s) that an
2334 indication is designed to report; see 7.3.29.2. Event definition query statements are not represented in
2335 CIM; thus there is no requirement for implementations or clients to interpret event definition query
2336 statements.

2337 PropertySet shall be "*", or a comma-separated list of property names.

2338 AlertIndicationClass shall be CIM_AlertIndication, or, if adaptations based on the
2339 AlertIndication adaptation adapt a class derived from CIM_AlertIndication, shall be replaced by the name
2340 of the adapted alert indication class.

2341 OwningEntity shall be the name of the organization defining the alert indication. In profiles owned by
2342 DMTF, the value shall be "DMTF".

2343 MessageIdentification shall identify each referenced alert message, as required by 7.3.31.3.

2344 Referencing profiles in their adaptations based on the AlertIndication adaptation may refine the event
2345 definition; however, such refinements shall remain within the constraints established by the query
2346 statement specified in this subclause.

2347 If a referencing profile defining an adaptation based on the AlertIndication adaptation does not require
2348 refining the query statement specified in this subclause, then a repetition of the query statement is not
2349 required as part of the adaptation in the referencing profile, and compliance with this subclause is
2350 achieved through designating a related alert message as required in 7.3.31.3.

2351 `AdditionalSelectionExpression` shall be a constant string that defines a selection expression
2352 conformant with the rules for selection expressions defined by [DSP0202](#). For example, the value of the
2353 `PerceivedSeverity` property could be constrained to specific values.

2354 7.3.31.3 Related alert messages

2355 Referencing profiles defining adaptations based on the `AlertIndication` adaptation as part of their alert
2356 indication adaptation shall reference one or more related CIM alert message(s) that are defined in a
2357 message registry conformant to [DSP0228](#).

2358 The formal requirements for referencing alert messages through message identifications as part of
2359 adaptation definitions are detailed in [DSP1001](#); as defined there, the main elements of a message
2360 identification are the name of the registry reference referring to the registry defining the alert message,
2361 and the message id as the concatenation of the value of the `PREFIX` attribute and the
2362 `SEQUENCE_NUMBER` attribute from the `MESSAGE_ID` element that defines the message within the
2363 message registry.

2364 CIM alert messages provide for a formalized and widely self-contained approach to define alert
2365 indications. CIM alert messages are defined in message registries. A message registry is an XML
2366 document that contains message definitions. [DSP0228](#) defines an XML schema for message registries.
2367 The schema defines the XML elements that can be used for message definitions. Each element is
2368 formally defined using the XML schema language. Each of these element definitions is annotated with
2369 documentation that may define formal requirements for the use of the message element.

2370 Each message definition in a message registry consists of a standard message identifier and a
2371 description of static and dynamic message elements and of other message components; for details, see
2372 [DSP0228](#).

2373 The `MESSAGE_ID` element within the message definition identifies the message within the scope of the
2374 message registry through a prefix and a sequence number.

2375 The `MESSAGE_DESCRIPTION` element within an alert message definition contains a plain text description
2376 of the event that is reported by the defined alert message. A profile modeling an alert indication shall rely
2377 on the event definition provided in the alert message description. In case the alert-message-based
2378 definition of the event is insufficient in the context of the profile, the profile may augment the event
2379 definition within its definition of the alert indication; however, the amendments to the event definition
2380 stated in a profile shall remain within the constraints defined by the event definition in the alert message
2381 definition in the message repository.

2382 The `<MESSAGE_COMPONENTS>` element within an alert message definition defines a sequence of static
2383 and dynamic elements that together compose the message. The static elements define constant text
2384 parts of the message. The dynamic elements reference property values in identified CIM instances, such
2385 that the property values become dynamic parts of the alert message.

2386 7.3.31.4 Indication origin

2387 If the alert indication is related to a managed object, and the CIM representation of that managed object is
2388 referenced by the value of the `AlertingManagedElement` property in the CIM representation of the alert
2389 indication, then the indication origin as required by 7.3.29.3 should be the namespace in which the CIM
2390 representation of that managed object exists.

2391 7.3.31.5 Element requirements

2392 7.3.31.5.1 General

2393 Table 46 lists the element requirements for the `AlertIndication` adaptation.

2394

Table 46 – AlertIndication: Element requirements

Elements	Requirement	Description
Base adaptations		
BasicIndication	Mandatory	See 7.3.29.
ReliableIndication	Conditional	Condition: The ReliableIndications feature (see 7.2.4) is implemented. See 7.3.30; note that this is a WBEM server related implementation requirement; see 7.1.
Properties		
AlertingElementFormat	Mandatory	Value shall match 2 (CIMObjectPath)
AlertingManagedElement	Mandatory	See 7.3.31.5.2.
AlertType	Mandatory	See 7.3.31.5.3.
Message	Optional	See 7.3.31.5.4.
MessageID	Mandatory	See 7.3.31.5.5.
OtherAlertType	Conditional	Condition: The AlertType property can have the value 1 (Other). Value shall be non-Null if the value of the AlertType property is 1 (Other).
OwningEntity	Mandatory	See 7.3.31.5.6.
PerceivedSeverity	Mandatory	See 7.3.31.5.7.
ProbableCause	Mandatory	See CIM schema definition.
ProbableCauseDescription	Conditional	Condition: The ProbableCause property can have the value 1 (Other). Value shall be non-Null if the value of the ProbableCause property is 1 (Other).
SystemName	Mandatory	See 7.3.31.5.8.
MessageArguments[]	Mandatory	See 7.3.31.5.9.

2395 7.3.31.5.2 Property: AlertingManagedElement

2396 If the managed element for which the alert indication is reported is represented by one or more CIM
 2397 instances within the implementation, then the value of the AlertingManagedElement property shall identify
 2398 the most prominent of these CIM instances, using the format of a WBEM-URI-UntypedInstancePath (as
 2399 defined in [DSP0207](#)); otherwise the value of the AlertingManagedElement property shall be Null.

2400 7.3.31.5.3 Property: AlertType

2401 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the AlertType property in
 2402 CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2403 content of the ALERT_TYPE element from the alert message definition in the message registry.

2404 7.3.31.5.4 Property: Message

2405 The requirement level of the Message property is optional.

2406 The Message property may contain the formatted alert message from the registry.

2407 **7.3.31.5.5 Property: MessageID**

2408 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the MessageID property in
 2409 CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2410 concatenation of the PREFIX and SEQUENCE_NUMBER attribute values from the alert message definition
 2411 in the message registry (that is, no further padding or adjustment of these values takes place).

2412 NOTE The SEQUENCE_NUMBER attribute value is not to be confused with the sequence number within a
 2413 sequence identifier that enables unique identification of the indications originating from a particular WBEM
 2414 server to a particular WBEM listener; see 7.4.2.

2415 **7.3.31.5.6 Property: OwningEntity**

2416 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the OwningEntity property
 2417 in CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2418 content of the OWNING_ENTITY element from the alert message definition in the message registry.

2419 **7.3.31.5.7 Property: PerceivedSeverity**

2420 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the PerceivedSeverity
 2421 property in CIM_AlertIndication instances conveying an alert message from a message registry to be set
 2422 to the content of the PERCEIVED_SEVERITY element from the alert message definition in the message
 2423 registry.

2424 **7.3.31.5.8 Property: SystemName**

2425 If the managed element for which the alert indication is reported is represented by a CIM instance within
 2426 the implementation, and the managed element is a component of a system that is represented by a
 2427 CIM_System instance, then the value of the SystemName property in the AlertIndication instance shall be
 2428 identical with the value of the Name property in the CIM_System instance; otherwise, the value of the
 2429 SystemName property shall be Null.

2430 **7.3.31.5.9 Property: MessageArguments[]**

2431 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the (string typed) MessageArguments
 2432 array property in CIM_AlertIndication instances conveying an alert message from a message registry to
 2433 contain one array entry for each dynamic element defined in the alert message, in the order specified by
 2434 the alert message definition in the message registry, where the value of the array element provides the
 2435 value of the dynamic element.

2436 If for a particular alert indication defined by a referencing profile the definition of a dynamic element
 2437 (including its description) within an alert message definition in a message registry is not sufficient to
 2438 identify a particular CIM instance and property as required by the referencing profile, then the referencing
 2439 profile shall specify augmenting provisions that explicitly identify an instance and a property that are
 2440 compatible with the definition of the dynamic element within the alert message.

2441 For example, assume that an alert message is defined in a message repository, as follows:

```
2442 <MESSAGE NAME="System state change">
2443   <MESSAGE_ID PREFIX="SVPC" SEQUENCE_NUMBER="0123" />
2444   <MESSAGE_DESCRIPTION>
2445     This message describes a system state change.
2446   </MESSAGE_DESCRIPTION>
2447   <MESSAGE_COMPONENTS>
2448     <STATIC_ELEMENT>The system </STATIC_ELEMENT>
2449     <DYNAMIC_ELEMENT NAME="SystemElementName"
2450       SOURCE_PROPERTY="CIM_System.ElementName" DATATYPE="string" />
2451     <STATIC_ELEMENT> changed its state to </STATIC_ELEMENT>
```

```

2452     <DYNAMIC_ELEMENT NAME="SystemState"
2453         SOURCE_PROPERTY="CIM_System.EnabledState" DATATYPE="string" />
2454     <STATIC_ELEMENT> .</STATIC_ELEMENT>
2455 </MESSAGE_COMPONENTS>
2456 <FIXED_MESSAGE_INSTANCE_VALUES TYPE="ALERT">
2457     <!-- . . . -->
2458 </FIXED_MESSAGE_INSTANCE_VALUES>
2459     <!-- . . . -->
2460 </MESSAGE>

```

2461 An Example System Virtualization profile might model an indication reporting state changes of both host
 2462 systems and virtual systems. In both cases the SVPC0123 alert message would be used, but the
 2463 identification of affected instances would need to be specialized separately for each case.

2464 Assuming that the profile defines a HostSystem adaptation of the CIM_System class for the
 2465 representation of host systems, and defines a HostStateChange indication adaptation in order to report
 2466 state changes of host systems, the requirements for the MessageArguments[] array property as part of
 2467 the HostStateChange indication adaptation would need to augment the alert message definition from the
 2468 message registry, as follows:

- 2469 • The value of MessageArguments[0] shall be the value of the ElementName property of the
 2470 HostSystem instance representing the host system that changed its state.
- 2471 • The value of MessageArguments[1] shall be the new value of the EnabledState property of the
 2472 HostSystem instance representing the host system that changed its state.

2473 7.3.31.6 Indication generation requirements

2474 The indication generation requirements of 7.3.29.5 apply respectively for the AlertIndication adaptation.

2475 7.3.32 LifecycleIndication: CIM_InstIndication

2476 7.3.32.1 General

2477 The LifecycleIndication adaptation models lifecycle indications of CIM instances; lifecycle indications are
 2478 described in 6.1.2.3.

2479 The LifecycleIndication adaptation adapts the CIM_InstIndication class and is based on the
 2480 BasicIndication adaptation (see 7.3.29); in addition, if the ReliableIndications feature (see 7.2.4) is
 2481 implemented, it is also based on the ReliableIndication adaptation (see 7.3.30).

2482 The implementation type of the LifecycleIndication indication adaptation is: "abstract".

2483 It is expected that the LifecycleIndication adaptation is used as a base adaptation for modeling lifecycle
 2484 indications in referencing profiles.

2485 7.3.32.2 Event definition requirements

2486 This subclause refines the event definition requirements established by the BasicIndication adaptation
 2487 (see 7.3.29.2) for the LifecycleIndication adaptation.

2488 Recall that lifecycle indication reports secondary events (see 6.1.1). The secondary event that is reported
 2489 by LifecycleIndication instances shall be described by an event definition query statement that conforms
 2490 to the following ABNF rule:

```

2491     "SELECT" WS PropertySet WS "FROM" WS LifecycleIndicationClass WS
2492     "WHERE" WS "ISA" WS ModelElement [ WS "WHERE" SelectionExpression ]

```

2493 PropertySet shall be "*", or a comma-separated list of property names.

2494 LifecycleIndicationClass shall be one of CIM_InstCreation, CIM_InstDeletion, or
 2495 CIM_InstModification, or a subclass of these indication classes.

2496 ModelElement shall identify a class for that the referencing profile defines a class adaptation, and for
 2497 which the modeled lifecycle indication reports secondary events. The class adaptation of that class shall
 2498 be stated as part of the description of the lifecycle indication adaptation in the referencing profile.

2499 NOTE For examples that comply with this requirement, see 7.3.33 and 7.3.34.

2500 SelectionExpression shall be a constant string that defines a selection expression conformant with
 2501 the rules for selection expressions defined by [DSP0202](#).

2502 NOTE These rules provide for referencing profiles being able to define one lifecycle indication for one target
 2503 adaptation per lifecycle indication adaptation. If for a particular target adaption a referencing profile intends
 2504 to model lifecycle indications for different lifecycle events (such as the creation, destruction or modification
 2505 of instances of the target adaptation), for each of these lifecycle events separate lifecycle indication
 2506 adaptations are required. Furthermore, if lifecycle indications are to be modeled for different target
 2507 adaptations, for each target adaptation separate lifecycle indication adaptations are required. As usual, if
 2508 common requirements exist for such lifecycle indication adaptations, these can be defined in a common
 2509 abstract base adaptation that is used as a base for the specific lifecycle indication adaptations, thereby
 2510 avoiding the repetition of the commonalities.

2511 **7.3.32.3 Indication origin**

2512 The indication origin as required by 7.3.29.3 shall be the namespace of the CIM instance referenced by
 2513 the value of the SourceInstanceModelPath property (see 7.3.32.4.3).

2514 **7.3.32.4 Element requirements**

2515 **7.3.32.4.1 General**

2516 Table 47 lists the element requirements for the LifecycleIndication adaptation.

2517 **Table 47 – LifecycleIndication: Element requirements**

Elements	Requirement	Description
Base adaptations		
BasicIndication	Mandatory	See 7.3.29.
ReliableIndication	Conditional	Condition: The ReliableIndications feature (see 7.2.4) is implemented. See 7.3.30; note that this is a WBEM server related implementation requirement; see 7.1.
Properties		
SourceInstance	Mandatory	See 7.3.32.4.2.
SourceInstanceModelPath	Mandatory	See 7.3.32.4.3.

2518 **7.3.32.4.2 Property: SourceInstance**

2519 The value of the SourceInstance property shall be an embedded instance of the class selected in the
 2520 query statement defining the event. The embedded instance shall be a copy of the instance for which the
 2521 lifecycle indication is reported. If the query statement specifies a specific selection of properties (other
 2522 than " * "), then the set of properties contained in the embedded instance shall be limited to those
 2523 selected; otherwise, the embedded instance shall at least contain values for each of the properties
 2524 required by the related adaptation of the selected class in the same referencing profile; see 7.3.29.2.

2525 **7.3.32.4.3 Property: SourceInstanceModelPath**

2526 The value of the SourceInstanceModelPath property shall refer to the same instance that is copied as an
 2527 embedded instance through the value of the SourceInstance property.

2528 **7.3.32.5 Indication generation requirements**

2529 The indication generation requirements of 7.3.29.5 apply respectively for the LifecycleIndication
 2530 adaptation.

2531 **7.3.33 ListenerDestinationRemovalIndication: CIM_InstDeletion**

2532 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2533 The ListenerDestinationRemovalIndication adaptation models a lifecycle indication that reports the
 2534 destruction of a CIM_ListenerDestination instance, as modeled in this profile by the ListenerDestination
 2535 adaptation (see 7.3.23). The destruction of a ListenerDestination instance is a secondary event caused
 2536 by the destruction of the represented listener destination; see 6.4.5.

2537 The requirement level of the ListenerDestinationRemovalIndication indication adaptation is optional.

2538 The implementation type of the ListenerDestinationRemovalIndication indication adaptation is:
 2539 "indication".

2540 Table 48 lists the element requirements for the ListenerDestinationRemovalIndication adaptation.

2541 **Table 48 – ListenerDestinationRemovalIndication: Element requirements**

Elements	Requirement	Description
Base adaptations		
LifecycleIndication	Mandatory	See 7.3.32.

2542 The requirement level of the ListenerDestinationRemovalIndication adaptation is optional.

2543 The event reported by the ListenerDestinationRemovalIndication adaptation is defined by the following
 2544 event definition query statement:

```
2545     SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA
2546     CIM_ListenerDestination
```

2547 **7.3.34 SubscriptionRemovalIndication: CIM_InstDeletion**

2548 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2549 The SubscriptionRemovalIndication adaptation models a lifecycle indication that reports the destruction of
 2550 a CIM_AbstractIndicationSubscription instance, as modeled in this profile by the AbstractSubscription
 2551 adaptation (see 7.3.25). The destruction of a CIM_AbstractIndicationSubscription instance is a secondary
 2552 event caused by the destruction of the represented subscription; see 6.1.1.

2553 The requirement level of the SubscriptionRemovalIndication indication adaptation is optional.

2554 The implementation type of the SubscriptionRemovalIndication indication adaptation is: "indication".

2555 Table 49 lists the element requirements for the SubscriptionRemovalIndication adaptation.

2556

Table 49 – SubscriptionRemovalIndication: Element requirements

Elements	Requirement	Description
Base adaptations		
LifecycleIndication	Mandatory	See 7.3.32.

2557 The requirement level of the SubscriptionRemovalIndication adaptation is optional.

2558 The event reported by the SubscriptionRemovalIndication adaptation is defined by the following query
 2559 statement:

```
2560     SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA
2561     CIM_AbstractIndicationSubscription
```

2562 **7.4 Reliable indication delivery**

2563 **7.4.1 General**

2564 This subclause defines mechanisms for the reliable delivery of indications from an implementation to a
 2565 listener as described in 6.1.5.

2566 Implementations implementing the ReliableIndications feature (see 7.2.4) shall comply with the
 2567 requirements specified in 7.4.3; note that in addition the requirements of the ReliableIndications
 2568 adaptation (see 7.3.30) apply.

2569 Implementations not implementing the ReliableIndications feature are not required to comply with the
 2570 provisions in this subclause or those in 7.3.30.

2571 Listeners implementing the ReliableIndications feature (see 7.2.4) shall comply with the provisions stated
 2572 in 7.4.4. Listeners not implementing the ReliableIndications feature are not required to comply with these
 2573 provisions and may ignore the sequence identifiers in received indications, as exposed by the values of
 2574 the SequenceContext and SequenceNumber properties in any received CIM_Indication instances.

2575 **7.4.2 Sequence identifier and sequence identifier lifetime**

2576 This subclause defines the concepts of *sequence identifier* and *sequence identifier lifetime*.

2577 The *sequence identifier* within an indication enables unique identification of the indications originating
 2578 from a particular WBEM server to a particular WBEM listener.

2579 A sequence identifier is composed of a sequence context and a sequence number.

2580 NOTE The sequence number within a sequence identifier is not to be confused with the SEQUENCE_NUMBER
 2581 attribute value that is part of the identification of the alert message that defines an alert indication; see
 2582 7.3.31.5.5.

2583 The sequence context is required to be unique for each listener destination maintained by the indication
 2584 service within a WBEM server; within that context the sequence number is required to be unique for each
 2585 indication delivered from the WBEM server to the listener referenced by the listener destination. The
 2586 requirements for the CIM representation of the sequence identifier in reliable indications are defined in
 2587 7.3.30.

2588 The *sequence identifier lifetime* maintained by an implementation is a duration defined as follows:

2589
$$\text{sequence-identifier-lifetime} = \text{number-of-retry-attempts} * \text{delivery-retry-interval} * 10$$

2590 In this formula the number-of-retry-attempts is the number of retry attempts as indicated by the value of
 2591 the DeliveryRetryAttempts property (see 7.3.2.3.3) in the IndicationService instance representing the

2592 indication service within the implementation, and the delivery-retry-interval is the duration of the delivery
2593 retry interval as indicated by the value of the DeliveryRetryInterval property (see 7.3.2.3.4) in the same
2594 instance.

2595 Within the sequence identifier lifetime an implementation that is implementing reliable indications may
2596 attempt to retry failed indication delivery attempts, as detailed in 7.4.3, and a listener implementing
2597 reliable indications may expect the delivery of anticipated indications, as detailed in 7.4.4.

2598 **7.4.3 WBEM server requirements**

2599 **7.4.3.1 General**

2600 Indication delivery is based on a publish/subscribe event paradigm, where an implementation delivers
2601 indications to subscribed listeners. The indication delivery may fail for various reasons, including
2602 unavailability of the listener or network issues. This subclause describes the requirements for the
2603 implementation that are related to reliable indication delivery. The mechanisms to deliver indications and
2604 to determine success or failure of indication delivery are protocol dependent; see the specifications of
2605 applicable protocols that specify mechanisms for indication delivery.

2606 **7.4.3.2 Prohibition of indication delivery for disabled or removed subscriptions**

2607 If a subscription is disabled or has been removed, the implementation should discard any undelivered
2608 indications for that subscription. For example, this applies if the implementation has queued indications
2609 for delivery retry, and the subscription is removed by a client before the delivery retry is executed.

2610 **7.4.3.3 Prohibition of repeated indication delivery**

2611 After an implementation has successfully delivered an indication to a listener, it shall not deliver that
2612 indication again to that same listener.

2613 **7.4.3.4 Requirements for the retry of failed indication deliveries**

2614 If the attempt to deliver an indication to a particular listener fails, the implementation shall retry the
2615 indication delivery as detailed in this subclause.

- 2616 1) The implementation shall wait for the duration of the delivery retry interval, as exposed by the
2617 value of the DeliveryRetryInterval property in the IndicationService instance (see 7.3.2)
2618 representing the indication service within the implementation.
- 2619 2) If the actual number of retry attempts is less than the maximum number of retry attempts as
2620 exposed by the value of the DeliveryRetryAttempts property in the IndicationService instance
2621 representing the indication service within the implementation, and the elapsed time after the first
2622 delivery is less than the sequence identifier lifetime as defined in 7.4.2, the implementation shall
2623 retry the failed indication delivery.
 - 2624 • If the retry is successful, delivery of that indication to the particular listener is complete.
 - 2625 • If the retry is not successful, and preconditions of step 2) still apply, then the
2626 implementation shall re-iterate starting with step 1).
 - 2627 • Otherwise, the indication shall be considered as not deliverable to the particular listener,
2628 and the requirements defined in 7.4.3.5 apply.

2629 **7.4.3.5 Requirements for undeliverable indications**

2630 This subclause defines the implementation behavior if an indication has been considered unable to be
2631 delivered to a listener, as described in 7.4.3.4.

2632 If the listener destination referencing that listener is permanent (see 7.3.23.3.3), the implementation shall
2633 record an error and shall no longer attempt to deliver that indication to that listener (that is, the
2634 implementation shall discard it). This action does not modify the listener destination and any of its
2635 subscriptions.

2636 If the listener destination referencing that listener is transient (see 7.3.23.3.3), the implementation shall
2637 record an error and shall no longer attempt to deliver that indication to that listener (that is, the
2638 implementation shall discard it). In addition, the listener destination and its subscriptions may be removed
2639 from the implementation as described in 7.4.3.6.

2640 **7.4.3.6 Requirements for the implicit removal of subscriptions and listener destinations**

2641 An implementation may remove a subscription and the referenced listener destination if the delivery of
2642 one or more indications to the represented listener failed as described in 7.4.3.4 and 7.4.3.5.

2643 The implementation behavior with respect to the implicit removal of subscriptions and listener destinations
2644 shall be exposed by the value of the SubscriptionRemovalAction property in the IndicationService
2645 instance representing the responsible indication service; see 7.3.2.3.5.

2646 **7.4.3.7 Behavior related to WBEM server restarts**

2647 Indications that have been generated but not yet delivered may get lost during a WBEM server crash or
2648 restart because there is not requirement to persist such indications.

2649 If the implementation chooses an algorithm for the construction of the sequence context part of the
2650 sequence identifier (see 7.4.2) that includes the WBEM server startup time, the potential re-use of the
2651 same sequence identifier is implicitly avoided. That way listeners can deal with indication delivery failures
2652 caused by WBEM server restarts in the same way they deal with other kinds of indication delivery failures.

2653 **7.4.4 WBEM listener requirements**

2654 **7.4.4.1 General**

2655 A listener shall keep track of each distinct sequence identifier of any indications received from a particular
2656 indication service for the duration of the sequence identifier lifetime maintained by that indication service,
2657 counting from the last time that sequence identifier was detected in a received indication from that
2658 indication service. If the same sequence identifier is used by two different indication services (for
2659 example, in two different implementations), the listener shall keep track of them independently.

2660 After the lifetime of a sequence identifier expires, the listener should discard the knowledge about that
2661 sequence identifier from that indication service. After the knowledge about a sequence identifier for an
2662 indication service has been discarded by the listener, a new usage of that sequence identifier in an
2663 indication from that indication service shall be treated by the listener like a new, unknown sequence
2664 identifier from that indication service.

2665 Keeping track of sequence identifiers in listeners enables the detection of lost and duplicate deliveries,
2666 and the detection and re-ordering of indications arriving out of order, as described in 7.4.4.5. Discarding
2667 the knowledge about sequence identifiers minimizes the resource requirements of the listener.

2668 **7.4.4.2 Determination of the expected sequence identifier of the next indication**

2669 From the sequence identifier of the last indication received from a particular implementation, a listener
2670 shall infer the expected sequence identifier of the next indication by incrementing the sequence number
2671 by 1, wrapping to an initial value of 0 if the maximum limit has been reached, and maintaining the
2672 sequence context.

2673 7.4.4.3 Lost indications

2674 If the sequence identifier of the next received indication sent from the same implementation does not
2675 match the expected value as described in 7.4.4.2, the listener shall consider the expected indication as a
2676 candidate for a lost indication. After waiting for the sequence identifier lifetime period as maintained by
2677 the implementation sending that indication, the listener shall conclude that the expected indication is lost.

2678 7.4.4.4 Duplicate indications

2679 Any additional indications received from the same implementation with the same sequence identifier shall
2680 be considered duplicates. In this case, the lifetime for the sequence identifier shall be adjusted starting
2681 with the delivery time of the most recently received duplicate indication, and adding the sequence
2682 identifier lifetime period as maintained by the implementation sending that indication.

2683 7.4.4.5 Out-of-order indications

2684 A listener that intends to re-establish the original order of indications before processing them needs to
2685 defer the processing of any prematurely arriving indication that does not have the expected sequence
2686 number, until the decision can be made as to whether the expected indications are lost.

2687 If the sequence identifier of the next received indication does not match the expected sequence identifier
2688 as described in 7.4.4.2, the listener shall cache such prematurely arriving indications and wait for delivery
2689 of the indication with the expected sequence identifier for a period of time defined by the sequence
2690 identifier lifetime (as defined in 7.4.4.1) of the last received indication from the same implementation.

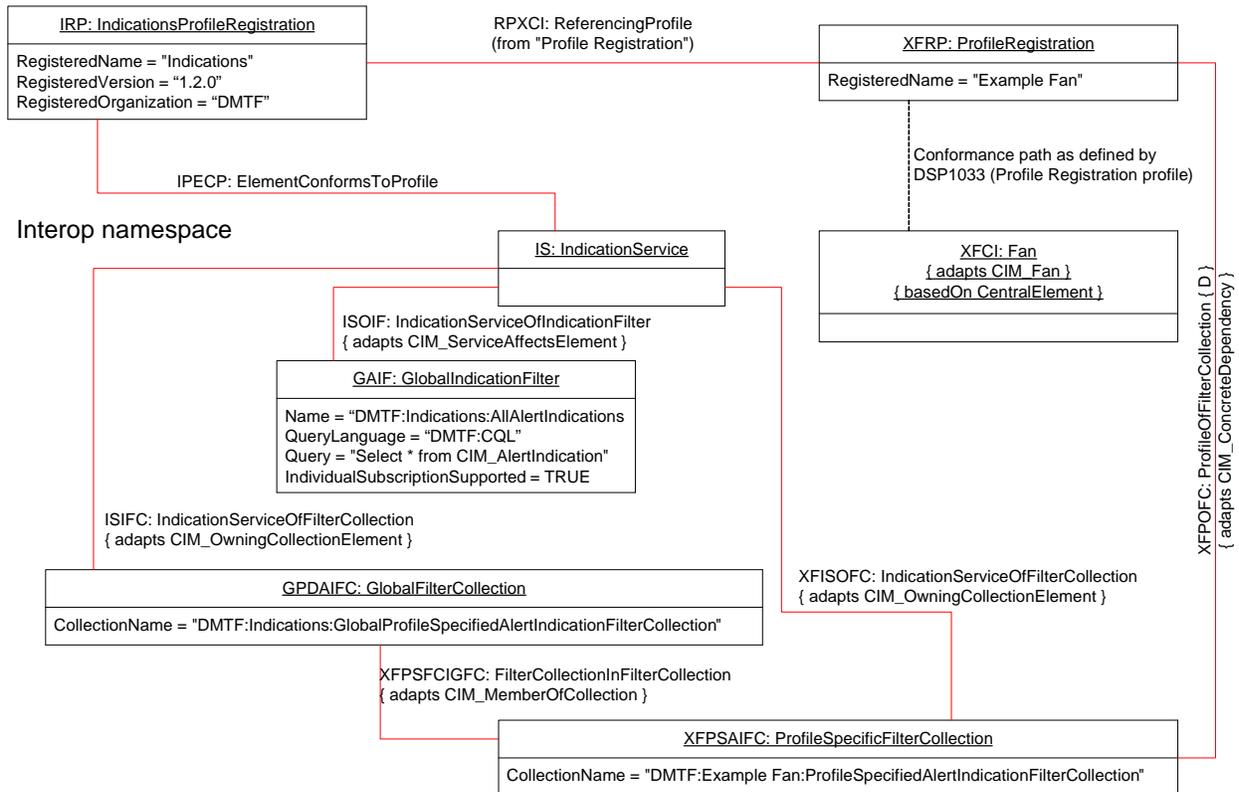
2691 If the indication with the expected sequence identifier is not received during that period, the expected
2692 indication should be considered lost (see 7.4.4.3).

2693 If the indication with the expected sequence identifier is received during that period, the indication order
2694 shall be re-ordered using their sequence numbers, such that the indications are processed in the order
2695 they were sent by the implementation.

2696 **8 Use Cases**

2697 **8.1 Object Diagrams**

2698 Figure 4 depicts a DMTF object diagram. It shows CIM instances exposed by the implementation of an
 2699 Example Fan profile that defines some indications (not shown in the diagram), and thus is required by
 2700 [DSP1001](#) to reference this profile, implying the implementation of respective elements defined in this
 2701 profile.



2702
 2703 **Figure 4 – DMTF object diagram: Global and profile-specific filter collections**

2704 The implemented version of this profile is represented by the RegisteredProfile instance IRP, the
 2705 implemented version of the Example Fan profile is represented by RegisteredProfile instance XFRP, and
 2706 the reference relationship is shown by the ReferencingProfile association instance RPXCI.

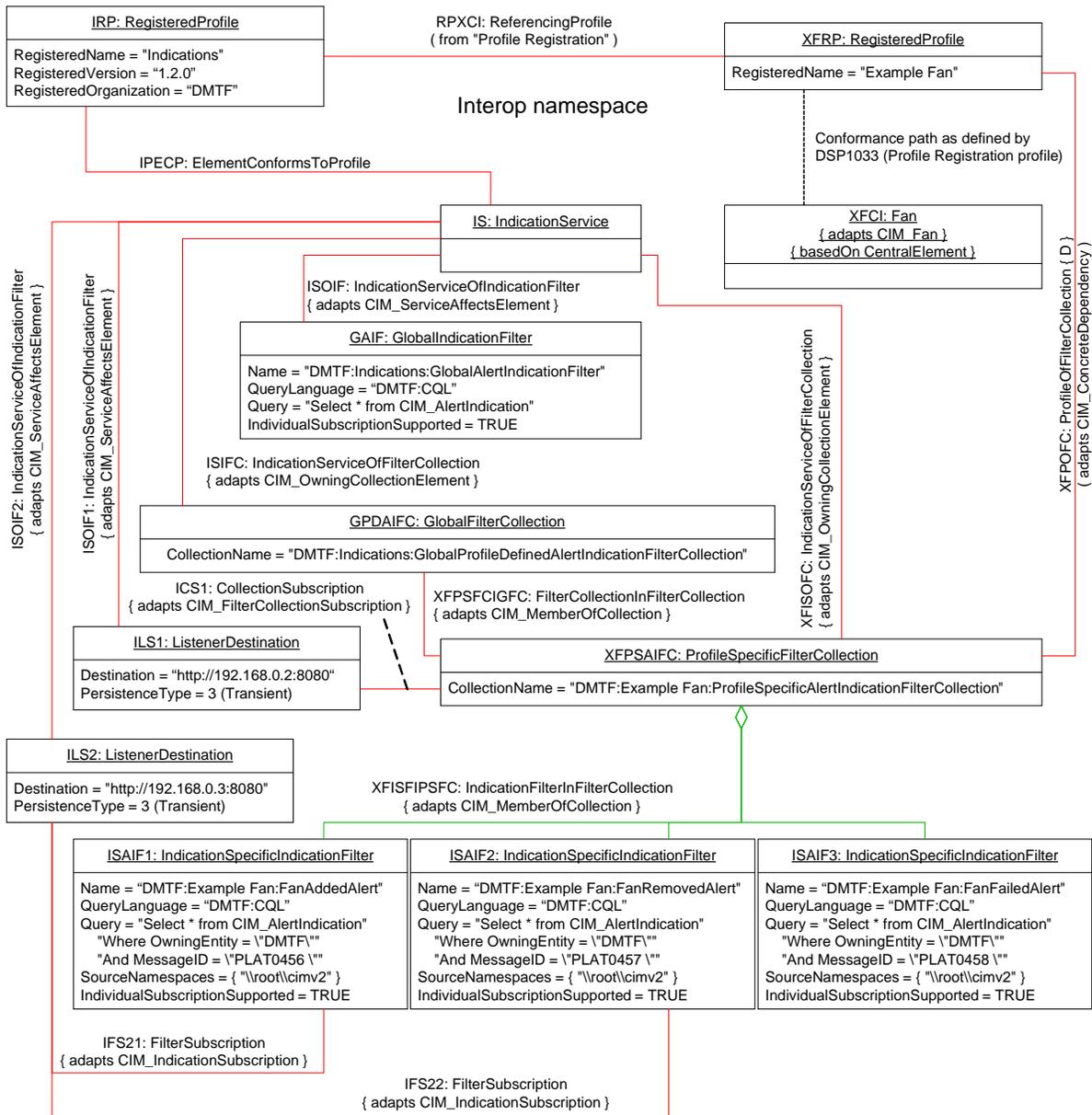
2707 The implementation of this profile exposes the IndicationService (see 7.3.2) instance IS representing the
 2708 implemented indication service. It also exposes the GlobalIndicationFilter (see 7.3.16) instance GAIF
 2709 representing the global indication filter covering all alert indications.

2710 Furthermore, the implementation of this profile exposes the GlobalFilterCollection (see 7.3.22) instance
 2711 GPDAIFC representing the global filter collection for alert indications with a defined coverage covering all
 2712 profile defined alert indications. The implementation of the Example Fan profile exposes the
 2713 ProfileSpecificFilterCollection (see 7.3.21) instance XFPSAIFC representing the related profile-specific
 2714 filter collection for alert indications with a defined coverage covering all alert indications defined in the
 2715 Example Fan profile.

2716 The global filter collection for alert indications represented by GPDAIFC contains the profile-specific filter
 2717 collection for alert indications represented by XFPSAIFC; this containment relationship is represented by
 2718 the FilterCollectionInFilterCollection (see 7.3.20) instance XFPSFCIGFC. Because the coverage of the

2719 global filter collection is explicitly represented by containment, in this case its coverage is inspectable by
2720 clients. However, the CIM representation of the contained profile-specific filter collection for alert
2721 indications represented by XFPSAIFC does not expose any contained elements. In that case clients
2722 would require prior knowledge of the defined coverage, that is, all alert indications defined in the Example
2723 Fan profile, which (because of the explicitly represented containment relationship) is in this example also
2724 the coverage of the global filter collection for alert indications represented by GPDAIFC.

2725 Figure 5 depicts a DMTF object diagram. It shows a variant of the situation illustrated in Figure 4.



rootcimv2 namespace

NOTE: The indications originate in this namespace, but do not exist in the namespace because they are transitional objects

XFALERT1: FanAddedAlert	XFALERT2: FamRemovedAlert	XFALERT3: FanFailedAlert
IndicationIdentifier = "XFALERT1" IndicationTime = "23:30:00 09/30/2009" OwingEntity = "DMTF" MessageID = "PLAT0456" AlertingManagedElement = "<URI referencing a CIM_Fan instance representing the added fan>" AlertType = 5 (Device Alert) PerceivedSeverity = 2 (Information)	IndicationIdentifier = "XFALERT2" IndicationTime = "23:45:00 09/30/2009" OwingEntity = "DMTF" MessageID = "PLAT0457" AlertingManagedElement = "<URI referencing a CIM_Fan instance that represented the removed fan>" AlertType = 5 (Device Alert) PerceivedSeverity = 3 (Degraded / Warning)	IndicationIdentifier = "XFALERT3" IndicationTime = "23:55:00 09/30/2009" OwingEntity = "DMTF" MessageID = "PLAT0458" AlertingManagedElement = "<URI referencing a CIM_Fan instance representing the failed fan>" AlertType = 5 (Device Alert) PerceivedSeverity = 4 (Minor)

2726

2727

Figure 5 – DMTF object diagram: Filter collections and contained indication filters

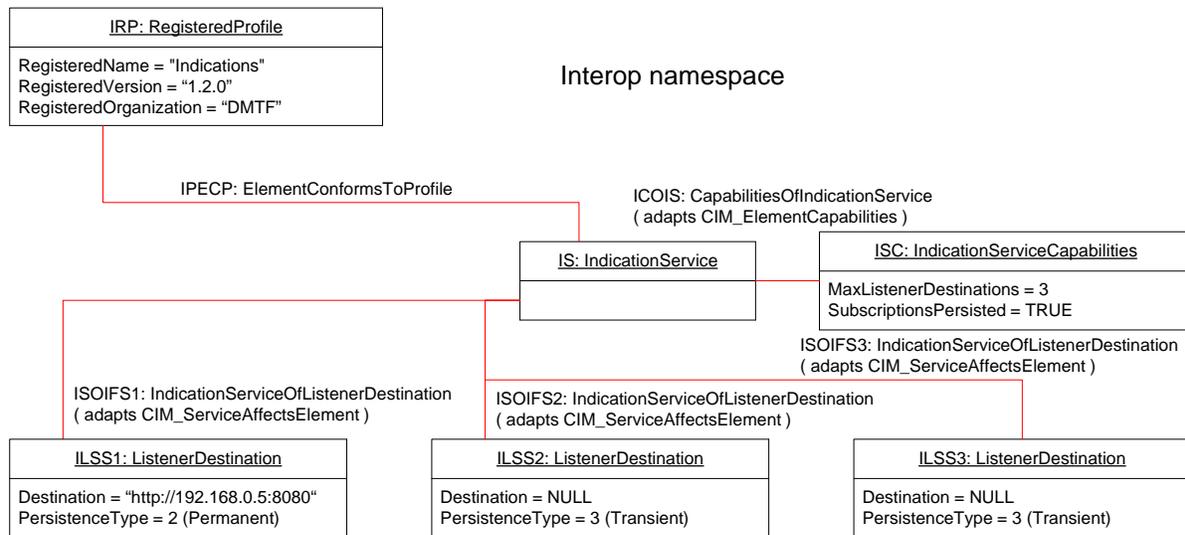
2728 The first difference from the situation shown in Figure 4 is that in Figure 5 the profile-specific filter
 2729 collection for alert indications represented by XFISAIFC contains three indication filters, represented by
 2730 the IndicationSpecificIndicationFilter instances ISAIF1, ISAIF2 and ISAIF3. Hence the coverage of the
 2731 profile-specific filter collection for alert indications represented by XFPSAIFC is now defined by the
 2732 contained indication filters, that is, it covers the three alert indications described by the alert messages
 2733 with the IDs PLAT0456, PLAT0457, and PLAT0458.

2734 It is important to recapture that — as with any indication gate — the presence of the CIM representation
 2735 of specific indication filters does not indicate that the covered indications are actually implemented. The
 2736 semantics of indication gates are defined with respect to *filtering*, but *not* with respect to *generating*,
 2737 indications (see 7.3.11.2 and 7.3.17.2). Thus, a subscribed listener is guaranteed only to be delivered any
 2738 *generated* indication that is within the coverage of the indication gate, but the *generation* of the indication
 2739 is not guaranteed. For that reason referencing profiles need to model other elements — such as
 2740 capabilities — for the purpose of conveying the information about which indications defined in the
 2741 referencing profile are actually implemented and thus generated when the respective event occurs; the
 2742 definition of such mechanisms is outside the scope of this profile.

2743 The second difference between Figure 4 and Figure 5 is that in Figure 5 listener destinations are
 2744 represented by the ListenerDestination instances ILS1 and ILS2. The listener referenced by ILS1 is
 2745 subscribed to the profile-specific filter collection represented by XFPSAIFC, and the listener referenced
 2746 by ILS1 is subscribed to the indication-specific indication filters represented by ISAIF1 and ISAIF2.

2747 Lastly, the representations of three indications are shown at the bottom of Figure 5, along with their origin
 2748 namespace. Each of these indications is within the coverage of the indication filter represented directly
 2749 above it. Thus, the alert indications represented by XFALERT1 and XFALERT2 are delivered to both the
 2750 listeners represented by ILS1 and ILS2, whereas XFALERT3 is only delivered to ILS1.

2751 Figure 6 depicts the DMTF object diagram for an implementation that supports a fixed number of listener
 2752 destinations.



2753

2754

Figure 6 – DMTF object diagram: Static listener destinations

2755 In the example shown in Figure 6, an implementation supports a maximum of three listener destinations,
 2756 indicated by the value of the MaxListenerDestinations property in the IndicationServiceCapabilities
 2757 instance ISC that describes the capabilities of the indication service within the implementation. The three
 2758 listener destinations are represented by the three respective ListenerDestination instances ILSS1, ILSS2,
 2759 and ILSS3. The listener destination represented by ILSS1 is currently configured as a permanent listener
 2760 destination, referencing the listener reachable under URI "http://192.168.0.5:8080". The listener

2761 destinations represented by ILSS2 and ILSS3 currently are free listener destinations as indicated by the
 2762 value Null for the Destination property, that is, they are not currently configured for a specific listener. A
 2763 client can request modifications of any of the listener destinations in order to reference a desired listener
 2764 for indication delivery by modifying the representing ListenerDestination instances.

2765 **8.2 LocateIndicationService: Locate the indication service provided by an** 2766 **implementation of this profile**

2767 **8.2.1 Preconditions**

2768 The client knows the following:

- 2769 • The identifying information of a WBEM server (for example, its IP address and the port number
 2770 if the WBEM server implements CIM operations over http as described in [DSP0223](#))
- 2771 • Name, required version, and registered organization of this profile as stated in 7.3.5

2772 **8.2.2 Flow of activities**

2773 1) The client obtains all IndicationsProfileRegistration instances (see 7.3.5), applying respective
 2774 use cases described in [DSP1033](#) to locate CIM_RegisteredProfile instances representing profile
 2775 registrations of particular profiles and selecting those instances where the values of the
 2776 RegisteredName, RegisteredVersion, and RegisteredOrganization properties match the
 2777 required input values.

2778 The result is zero or more IndicationsProfileRegistration instances (see 7.3.23).

2779 NOTE 1 Typically only one instance is returned, but if this profile is implemented more than once within the
 2780 identified WBEM server, more than one instance may be returned.

2781 If no instance was detected, this use case is complete and the client knows that the required
 2782 version of this profile is not implemented within the WBEM server. If one or more instances
 2783 were detected, any of them represents the required version of this profile, and the client can
 2784 select any of these for further processing.

2785 2) The client applies use cases described in [DSP1033](#) in order to locate instances of the
 2786 IndicationService adaptation that is the central class adaptation defined in this profile.

2787 The result is zero or one IndicationService instances (see 7.3.2).

2788 NOTE 2 Technically, more than one instance could be returned, but that would indicate a non-compliant
 2789 implementation of this profile.

2790 If no instance was detected, this use case is complete and the client knows that an indication
 2791 service is not presently active within the identified WBEM server. If one or more instances were
 2792 detected, any of them represents an indication service compliant to the requirements specified
 2793 in this profile, and the client can select any of these for further processing.

2794 **8.2.3 Postconditions**

2795 Unless errors occurred, the client either knows an IndicationService instance (including its object path)
 2796 representing an indication service within the identified WBEM server with a behavior compliant to the
 2797 requirements specified in this profile or knows that either this profile is not implemented within the
 2798 identified WBEM server or that no indication service is presently active within the identified WBEM server.

2799 **8.3 LocateProfileIndicationService: Locate the indication service responsible for**
2800 **delivering indications defined by a referencing profile**

2801 **8.3.1 Preconditions**

2802 The client knows the following:

- 2803
 - The ProfileRegistration instance (including its object path) representing the profile registration of
- 2804 the referencing profile

2805 **8.3.2 Flow of activities**

- 2806 1) For the input ProfileRegistration instance, find the IndicationsProfileRegistration instances (see
- 2807 7.3.5) associated through ReferencedProfile instances (see [DSP1033](#)) (for example, using the
- 2808 GetAssociatedInstancesWithPath() operation).

2809 The result is zero or one IndicationsProfileRegistration instances (see 7.3.5).

2810 NOTE 1 Technically, more than one instance could be returned, but that would indicate a non-compliant

2811 implementation of the referencing profile.

2812 If no instance was detected, this use case is complete and the client knows that the

2813 implementation of the referencing profile did not implement indications.

- 2814 2) For the IndicationsProfileRegistration instance obtained in step 1), find the IndicationService
- 2815 instances (see 7.3.2) associated through ElementConformsToProfile instances (see 7.3.6) (for
- 2816 example, using the GetAssociatedInstancesWithPath() operation).

2817 The result is zero or one IndicationService instances (see 7.3.2).

2818 NOTE 2 Technically, more than one instance could be returned, but that would indicate a non-compliant

2819 implementation of this profile.

2820 **8.3.3 Postconditions**

2821 Unless errors occurred, the client knows an IndicationService instance (including its object path)

2822 representing an indication service that is responsible for delivering indications defined by the referencing

2823 profile.

2824 **8.4 DetermineIndicationServiceCapabilities: Determine the capabilities of an**
2825 **indication service**

2826 **8.4.1 Preconditions**

2827 The client knows all of the following:

- 2828
 - a copy of the IndicationService instance (including its object path) representing the indication
- 2829 service within the implementation

2830 NOTE For example, that IndicationService instance could be obtained by applying the LocateIndicationService

2831 use case (see 8.2) or the LocateProfileIndicationService use case (see 8.3).

2832 **8.4.2 Flow of activities**

- 2833 1) Inspecting property values of the IndicationService instance (see 7.3.2.3), the client can already
- 2834 determine some aspects of the behavior of the represented indication service.

2835 For example, the value of the FilterCreationEnabled property indicates whether the support for
 2836 dynamic indication filters as modeled by the DynamicIndicationFilters feature (see 7.2.1) is
 2837 available.

2838 The values of the DeliveryRetryAttempts, the DeliveryRetryInterval, the
 2839 SubscriptionRemovalAction, and the SubscriptionRemovalTimeInterval indicate if and to what
 2840 extent the support for reliable indications as modeled by the ReliableIndications feature (see
 2841 7.2.4) is available.

2842 2) Find the IndicationsServiceCapabilities instance (see 7.3.7) representing the capabilities of the
 2843 input indication service, by traversing the CIM_ServiceAffectsElement association modeled by
 2844 the CapabilitiesOfIndicationService association adaptation (see 7.3.8) by invoking the
 2845 GetAssociatedInstancesWithPath() operation with the following actual values for the input
 2846 parameters:

- 2847 – InstanceName: the object path to the input IndicationService instance
- 2848 – AssocClass: "CIM_ElementCapabilities", the adapted class of the
 2849 CapabilitiesOfIndicationService association adaptation
- 2850 – ResultClass: "CIM_IndicationServiceCapabilities", the adapted class of the
 2851 IndicationServiceCapabilities adaptation

2852 The result is zero or one IndicationServiceCapabilities instance.

2853 NOTE Technically, more than one instance could be returned, but that would indicate a non-compliant
 2854 implementation of this profile.

2855 If an IndicationServiceCapabilities instance was returned, the use case continues with step 3);
 2856 otherwise, it continues with step 4).

2857 3) Inspect the property values of the returned IndicationServiceCapabilities instance (see 7.3.7).
 2858 The values of those properties with names ending with "IsSettable" enable the client to
 2859 determine whether client modification of respective aspects of the behavior of the input
 2860 indication service is possible. The values of the MaxListenerDestinations and the
 2861 MaxActiveSubscriptions properties expose the upper limits for the number of listener
 2862 destinations and for the number of subscriptions supported by the indication service, and the
 2863 value of the SubscriptionsPersisted property exposes whether subscriptions are persisted over
 2864 restarts of the input indication service. This step completes this use case.

2865 4) Continue here after step 2) if no IndicationServiceCapabilities instance was returned. In this
 2866 case, client modification of the indication service is not supported, and the upper limits for the
 2867 number of supported listener destinations and subscriptions is not exposed by the
 2868 implementation; in addition, whether subscriptions are persisted over indication service restarts
 2869 is not exposed.

2870 8.4.3 Postconditions

2871 Unless errors occurred, the client knows the capabilities of the input indication service as far as it is
 2872 exposed by the representing IndicationService instance, by the related IndicationServiceCapabilities
 2873 instance, and by initial behavior specified in this profile.

2874 8.5 ModifyIndicationService: Modify functional aspects of an indication service

2875 The client knows all of the following:

- 2876 • a copy of the IndicationService instance (including its object path) (see 7.3.2) representing the
 2877 indication service within the implementation (see the LocateIndicationService use case in 8.2)

- 2878 • a copy of the IndicationServiceCapabilities instance (including its object path) (see 7.3.7)
 2879 representing the capabilities of the indication service within the implementation (See the
 2880 DetermineIndicationServiceCapabilities use case in 8.4.)

2881 **8.5.1 Flow of activities**

- 2882 1) Inspect the property values in the input IndicationsServiceCapabilities instance (see 7.3.7)
 2883 representing the capabilities of the input indication service to determine which properties in the
 2884 IndicationService instance are modifiable. (See step 3) in the
 2885 DetermineIndicationServiceCapabilities use case in 8.4.)
- 2886 2) If admissible by the determination of step 1), in the input local copy of the input
 2887 IndicationService instance, modify property values as desired. For example, if the value of the
 2888 DeliveryRetryAttemptsIsSettable property in the IndicationServiceCapabilities instance is True,
 2889 a modification of the corresponding DeliveryRetryAttempts property in the IndicationService
 2890 instance is admissible.
- 2891 3) Use the ModifyInstance() operation to request the desired change in the behavior of the
 2892 indication service, providing the modified copy of the IndicationService instance as the actual
 2893 value of the ModifiedInstance parameter.

2894 **8.5.2 Postconditions**

2895 Unless errors occurred, the desired change of functional aspects of the input indication service is
 2896 effective.

2897 **8.6 ListListenerDestinations: List all listener destinations exposed by an 2898 implementation**

2899 **8.6.1 Preconditions**

2900 The client knows all of the following:

- 2901 • the object path to the IndicationService instance representing the indication service within the
 2902 implementation (see 8.2)

2903 **8.6.2 Flow of activities**

- 2904 1) Find all listener destinations within the responsibility of the indication service by traversing the
 2905 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfListenerDestination
 2906 adaptation (see 7.3.24) by invoking the GetAssociatedInstancesWithPath() operation with the
 2907 following actual values for the input parameters:
- 2908 – InstanceName: the object path to the input IndicationService instance
 - 2909 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
 2910 IndicationServiceOfListenerDestination adaptation
 - 2911 – ResultClass: "CIM_ListenerDestination", the adapted class of the ListenerDestination
 2912 adaptation
- 2913 The result is a set of ListenerDestination instances (see 7.3.23).

2914 **8.6.3 Postconditions**

2915 Unless errors occurred, the client knows all ListenerDestination instances (including their object paths)
 2916 representing all the listener destinations maintained by the implementation.

2917 **8.7 SelectListenerDestination: Select an existing listener destination referencing**
 2918 **a desired listener**

2919 **8.7.1 Preconditions**

2920 The client knows all of the following:

- 2921 • the object path to the IndicationService instance representing the indication service within the
2922 implementation (see 8.2)
- 2923 • the URI exposed by the desired listener
- 2924 • the particular protocol to be applied when delivering these indications

2925 **8.7.2 Flow of activities**

2926 1) Execute the ListListenerDestinations use case (see 8.6).

2927 The result is a set of ListenerDestination instances (see 7.3.23).

2928 2) Inspect each ListenerDestination instance resulting from step 1) by checking the value of the
2929 Destination property against the input URI, and by checking whether the value of the Protocol
2930 property matches the particular protocol for this use case.

2931 If both conditions are met, the located ListenerDestination represents a listener destination that
2932 within the implementation represents the particular listener, and this use case is complete;
2933 otherwise, the client needs to repeat step 2), inspecting further ListenerDestination instances
2934 from the result of step 1).

2935 3) If all result elements from step 1) checked in step 2) did not yield a ListenerDestination instance
2936 referencing the listener, then this use case is complete and the client knows that the listener is
2937 not presently represented by a listener destination within the implementation.

2938 **8.7.3 Postconditions**

2939 Unless errors occurred, the client either knows a ListenerDestination instance (including its object path)
2940 representing a listener destination within the implementation that references the particular listener, or
2941 knows that the listener is not referenced by any listener destination within the implementation.

2942 In the latter case, and if the implementation has also implemented the dynamic creation of listener
2943 destinations, the client could apply the CreateListenerDestination use case (see 8.8) to dynamically
2944 create a respective listener destination within the implementation that represents the desired listener.

2945 **8.8 CreateListenerDestination: Create a new listener destination**

2946 **8.8.1 Preconditions**

2947 The client knows all of the following:

- 2948 • The same as for the SelectListenerDestination use case; see 8.7.1.

2949 **8.8.2 Flow of activities**

2950 1) Execute the SelectIndicationFilter use case (see 8.7).

2951 If a listener destination referencing the desired listener is found, use that; in this case, this use
2952 case is complete.

- 2953 2) Prepare a local instance of the CIM_ListenerDestination class that complies with the
 2954 requirements of the ListenerDestination adaptation (see 7.3.23), inserting property values as
 2955 follows:
- 2956 – Destination: the identification of the listener that the new listener destination is to
 2957 reference, using the format required in 7.3.23.3.2. The format needs to be compatible
 2958 with the requested protocol.
 - 2959 – PersistenceType: the durability requested for the new listener destination, using the
 2960 format required in 7.3.23.3.3.
 - 2961 – Protocol: the protocol to used for the communication with the listener, using the format
 2962 required by the CIM schema definition of the CIM_ListenerDestination class.
- 2963 3) Request the creation of the new listener destination in the implementation by invoking the
 2964 CreateInstance() operation, providing the CIM_ListenerDestination instance prepared in step 2)
 2965 as the actual value of the NewInstance parameter.
- 2966 If successful, the operation returns the object path of the ListenerDestination instance
 2967 representing the newly created listener destination.
- 2968 If not successful, the operation returns a CIM status code providing details about the failure
 2969 (see 7.3.23.3.4).

2970 8.8.3 Postconditions

2971 Unless errors occurred, the client knows the object path of a ListenerDestination instance representing a
 2972 listener destination referencing the desired listener that either preexisted or was created; otherwise, the
 2973 client knows details about why it was not possible to find or dynamically create the respective listener
 2974 destination.

2975 8.9 FindFreeListenerDestination: Find a free listener destination

2976 8.9.1 Preconditions

2977 The client knows all of the following:

- 2978 • the object path to the IndicationService instance representing the indication service within the
 2979 implementation (see 8.2)

2980 8.9.2 Flow of activities

- 2981 1) Execute the ListListenerDestinations use case (see 8.6).
- 2982 The result of this step is the set of ListenerDestination instances (including their object paths)
 2983 representing all the listener destinations within the implementation.
- 2984 2) From the result of step 1), select a free listener destination; free listener destinations are
 2985 represented by those ListenerDestination instances where the value of the Destination property
 2986 is Null.

2987 8.9.3 Postconditions

2988 Unless errors occurred, the client knows a free listener destination, or knows that presently no free
 2989 listener destinations exist within the implementation.

2990 **8.10 ModifyListenerDestination: Modify an existing listener destination**

2991 **8.10.1 Preconditions**

2992 The client knows all of the following:

- 2993 • a local copy of a ListenerDestination instance (see 7.3.23)

2994 NOTE For example, the listener destination and its representing ListenerDestination instance might have been
2995 obtained by executing the FindFreeListenerDestination use case described in 8.9.

2996 **8.10.2 Flow of activities**

- 2997 1) Modify the local copy of the ListenerDestination instance, maintaining compliance with the
2998 requirements of the ListenerDestination adaptation (see 7.3.23).
- 2999 2) Modify the listener destination maintained by the implementation by invoking the
3000 ModifyInstance() operation, providing the CIM_ListenerDestination instance prepared in step 1)
3001 as the actual value of the ModifiedInstance parameter.

3002 If successful, the operation returns without error; otherwise, the operation returns a CIM status
3003 code providing details about the failure (see 7.3.23.3.6).

3004 **8.10.3 Postconditions**

3005 Unless errors occurred, the listener destination represented by the input ListenerDestination instance was
3006 modified; otherwise, the client knows details about why it was not possible to modify the represented
3007 listener destination.

3008 **8.11 DeleteListenerDestination: Delete an existing listener destination**

3009 **8.11.1 Preconditions**

3010 The client knows all of the following:

- 3011 • the object path to a ListenerDestination instance (see 7.3.23)

3012 **8.11.2 Flow of activities**

- 3013 1) For the input ListenerDestination instance, find all AbstractSubscription instances (see 7.3.25)
3014 referencing the ListenerDestination instance (for example, using the
3015 GetReferencingInstancePaths() operation).
- 3016 2) Delete all subscriptions referencing the input listener destination by executing the
3017 DeleteSubscription use case (see 8.21) for each AbstractSubscription instance returned by step
3018 1).
- 3019 3) Invoke the DeleteInstance() operation on the input ListenerDestination instance, effecting the
3020 deletion of the referenced listener destination.

3021 **8.11.3 Postconditions**

3022 Unless errors occurred, the input listener destination is deleted and no longer represented by any
3023 ListenerDestination instances.

3024 8.12 FindIndicationFilter: Find an indication filter covering a particular indication

3025 8.12.1 Preconditions

3026 The client knows all of the following:

- 3027 • the object path to the IndicationService instance representing the indication service within the
3028 implementation (see 7.3.2)
- 3029 • an implemented indication. Knowledge about whether or not a particular indication is actually
3030 implemented could for example be obtained by inspecting respective capabilities exposed by an
3031 implementation of a referencing profile that defines an adaptation of the particular indication.

3032 8.12.2 Flow of activities

3033 1) Find all indication filters within the responsibility of the indication service by traversing the
3034 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfIndicationFilter
3035 association adaptation (see 7.3.14) by invoking the GetAssociatedInstancesWithPath()
3036 operation with the following actual values for the input parameters:

- 3037 – InstanceName: the object path to the input IndicationService instance
- 3038 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
3039 IndicationServiceOfIndicationFilter association adaptation
- 3040 – ResultClass: "CIM_IndicationFilter", the adapted class of the IndicationFilter adaptation

3041 The result of this step is a set of IndicationFilter instances (see 7.3.11).

3042 2) Inspect each IndicationFilter instance resulting from step 1) by first checking the value of the
3043 QueryLanguage property. If the query language indicated by that value is interpretable by the
3044 client, interpret the query statement presented by the value of the Query property; otherwise,
3045 continue inspecting the next IndicationFilter instance returned by step 1).

3046 If the desired indication is not within the coverage as expressed by the query statement, then
3047 continue inspecting the next IndicationFilter instance returned by step 1).

3048 3) If the client desires to subscribe to the indication filter, continue by inspecting the IndicationFilter
3049 instance resulting from step 1) by checking whether the value of the
3050 IndividualSubscriptionSupported property is True. If so, this use case is complete; otherwise,
3051 continue with step 2) inspecting the next IndicationFilter instance returned by step 1); otherwise,
3052 this use case is complete.

3053 8.12.3 Postconditions

3054 Unless errors occurred, and if step 3) produced a suitable IndicationFilter instance, the client by that
3055 instance (including its object path) knows an indication filter that covers the desired indication and that
3056 supports individual subscriptions; otherwise, the client knows that within the responsibility of the indication
3057 service no such indication filter exists.

3058 8.13 DetermineQueryLanguages: Determine the set of query languages 3059 supported for query statements

3060 8.13.1 Preconditions

3061 The client knows all of the following:

- 3062 • The same as for the FindIndicationFilter use case described in 8.12.1.

3063 NOTE The procedure outlined in this use case is only an auxiliary approach to be pursued if preliminary
3064 knowledge about the query languages supported by an implementation is not available to the client.

3065 **8.13.2 Flow of activities**

- 3066 1) Execute steps 1) and 2) of the FindIndicationFilter use case (see 8.9), but vary step 2) to collect
3067 the query languages applied by all the inspected indication filters.

3068 **8.13.3 Postconditions**

3069 Unless errors occurred, the client knows all the query languages in use by existing indication filters.

3070 NOTE Because not all query languages supported by an implementation might be in use by indication filters, the
3071 set of query languages obtained by executing this use case is actually an open subset of the set of
3072 supported query languages.

3073 **8.14 CreateIndicationFilter: Create a dynamic indication filter covering a
3074 particular indication**3075 **8.14.1 Preconditions**

3076 The client knows all of the following:

- 3077 • The same as for the FindIndicationFilter use case described in 8.12.1.

3078 **8.14.2 Flow of activities**

- 3079 1) Execute the FindIndicationFilter use case (see 8.9).

3080 If a suitable indication filter covering the desired indication is found, use that; in this case, this
3081 use case is complete.

- 3082 2) If not already done previously, execute step 1) of the DetermineIndicationServiceCapabilities
3083 use case (see 8.4) and determine by the value of the FilterCreationEnabled property whether
3084 the support for dynamic indication filters as modeled by the DynamicIndicationFilters feature
3085 (see 7.2.1) is available.

- 3086 3) If the set of query languages supported by the implementation is not known a priori, execute the
3087 DetermineQueryLanguages use case (see 8.13).

- 3088 4) Prepare a local instance of the CIM_IndicationFilter class that complies with the requirements of
3089 the DynamicIndicationFilter adaptation (see 7.3.13), inserting property values as follows:

3090 – QueryLanguage: a query language supported by the implementation; see 7.3.11.3.6.

3091 – Query: the query statement covering the desired set of indications; see 7.3.11.3.5.

3092 NOTE Additional constraints on properties of the CIM_Indication class selected by the
3093 query statement may be specified through the WHERE clause; however, if the
3094 implementation is unable to comply with these constraints, the operation will fail.

3095 – SourceNamespaces[]: a list of local namespace names identifying the namespaces
3096 considered as ; see 7.3.11.3.3.

- 3097 5) Request the creation of the new dynamic indication filter in the implementation by invoking the
3098 CreateInstance() operation, providing the CIM_IndicationFilter instance prepared in step 4) as
3099 the actual value of the NewInstance parameter.

3100 If successful, the operation returns the object path of the DynamicIndicationFilter instance
3101 representing the newly created dynamic indication filter.

3102 If not successful, the operation returns a CIM status code providing details about the failure
3103 (see 7.3.13.2.2).

3104 8.14.3 Postconditions

3105 Unless errors occurred, the client knows the object path of an IndicationFilter instance representing an
3106 indication filter covering the desired indication that either preexisted or was dynamically created;
3107 otherwise, the client knows details about why it was not possible to find or dynamically create the
3108 respective indication filter.

3109 8.15 ModifyIndicationFilter: Modify a dynamic indication filter**3110 8.15.1 Preconditions**

3111 The client knows all of the following:

- 3112 • a local copy of an DynamicIndicationFilter instance (see 7.3.13)

3113 NOTE For example, that dynamic indication filter and its representing DynamicIndicationFilter instance might
3114 have been created by executing the CreateIndicationFilter use case; see 8.14.

3115 8.15.2 Flow of activities

- 3116 1) Modify the local copy of the DynamicIndicationFilter instance, maintaining compliance with the
3117 requirements of the DynamicIndicationFilter adaptation (see 7.3.13).
- 3118 2) Modify the dynamic indication filter maintained by the implementation by invoking the
3119 ModifyInstance() operation, providing the DynamicIndicationFilter instance prepared in step 1)
3120 as the actual value of the ModifiedInstance parameter.
- 3121 3) If successful, the operation returns without error; otherwise, the operation returns a CIM status
3122 code providing details about the failure (see 7.3.13.2.4).

3123 8.15.3 Postconditions

3124 Unless errors occurred, the dynamic indication filter represented by the input DynamicIndicationFilter
3125 instance was modified; otherwise, the client knows details about why it was not possible to modify the
3126 represented dynamic indication filter.

3127 8.16 DeleteIndicationFilter: Delete a dynamic indication filter**3128 8.16.1 Preconditions**

3129 The client knows all of the following:

- 3130 • the object path to a DynamicIndicationFilter instance (see 7.3.13)

3131 8.16.2 Flow of activities

- 3132 1) For the input DynamicIndicationFilter instance, find all AbstractSubscription instances (see
3133 7.3.25) referencing the DynamicIndicationFilter instance (for example, using the
3134 GetReferencingInstancePaths() operation).
- 3135 2) Delete all subscriptions referencing the input listener destination, by executing the
3136 DeleteSubscription use case (see 8.21) for each AbstractSubscription instance returned by step
3137 1).
- 3138 3) Invoke the DeleteInstance() operation on the input DynamicIndicationFilter instance, effecting
3139 the deletion of the referenced dynamic indication filter.

3140 8.16.3 Postconditions

3141 Unless errors occurred, the input dynamic indication filter is deleted and no longer represented by any
3142 DynamicIndicationFilter instances.

3143 8.17 CheckCollectionCoverage: Check the coverage of a filter collection

3144 8.17.1 Preconditions

3145 The client knows all of the following:

- 3146 • a local copy of a StaticFilterCollection instance (see 7.3.17), and the object path referencing the
3147 original StaticFilterCollection instance within the implementation

3148 8.17.2 Flow of activities

3149 1) Check whether the input filter collection contains any elements by resolving — from the
3150 StaticFilterCollection instance — the CIM_ConcreteComponent association as modeled by the
3151 IndicationFilterInFilterCollection association adaptation (see 7.3.19) and the
3152 FilterCollectionInFilterCollection association adaptation (see 7.3.20).

3153 If no contained elements are discovered, a defined coverage may apply as the coverage; in this
3154 case, skip to step 4).

3155 2) For each of the contained elements found in step 1), determine the contributed coverage and
3156 add that to the resulting aggregated coverage of the input filter collection.

3157 In the case of a contained indication filter, the contributed coverage is determined by inspecting
3158 the values of the QueryLanguage property and that of the Query property containing the query
3159 statement.

3160 In the case of a contained filter collection, the contributed coverage is determined by recursively
3161 applying this use case (8.17).

3162 3) Aggregate the contributed coverage of each contained element as determined in step 2) into the
3163 resulting aggregated coverage of the input filter collection. After completing this step the client
3164 knows the aggregated coverage of the input filter collection, and this use case is complete.

3165 4) This step applies if no contained elements were discovered in steps 2) and 3).

3166 Check the value of the CollectionName property in the StaticFilterCollection instance for the
3167 pattern required for the name the global filter collection covering all instance lifecycle
3168 indications, as detailed in 7.3.22.4.4.

3169 If the pattern matches, the client knows that the represented filter collection is the global filter
3170 collection covering all instance lifecycle indications; in this case, the client knows that the
3171 coverage of the input filter collection is all instance lifecycle indications and this use case is
3172 complete.

3173 5) Check the value of the CollectionName property in the StaticFilterCollection instance for the
3174 pattern required for the name of global filter collections for profile defined indications, as defined
3175 in 7.3.22.

3176 If the pattern matches, the client knows that the represented filter collection is a global filter
3177 collection for profile defined indications with a defined coverage as detailed in 7.3.22. The client
3178 needs to have a priori knowledge about the defined coverage of each referencing profile, and
3179 this use case is complete.

3180 6) Check the value of the CollectionName property in the StaticFilterCollection instance for the
3181 pattern required for the name of profile-specific filter collections as defined in 7.3.21.2.2.

3182 If the pattern matches, the client knows that the input filter collection is a profile-specific filter
3183 collection with a defined coverage as detailed in 7.3.21.3. The client needs to have a priori
3184 knowledge about the defined coverage of the identified referencing profile, and this use case is
3185 complete.

3186 7) If the input filter collection does not match any of the types determined in steps 4), 5), and 6),
3187 then no defined coverage applies. Furthermore, because no contained elements were
3188 discovered in step 2), the coverage of the input filter collection is empty (that is, it does not
3189 cover any indications).

3190 8.17.3 Postconditions

3191 Unless errors occurred, or in the cases determined in steps 5) and 6) above the client does not have a
3192 priori knowledge about the defined coverage(s), the client knows the coverage of the input filter collection.

3193 8.18 ObtainNamedCollection: Obtain a named filter collection

3194 8.18.1 Preconditions

3195 The client knows all of the following:

- 3196 • the object path to the IndicationService instance representing the indication service within the
3197 implementation (see 7.3.2)
- 3198 • the name of the named filter collection, for example, the name of a global filter collection or of a
3199 profile-specific filter collection

3200 8.18.2 Flow of activities

3201 1) Find all filter collections within the responsibility of the indication service by traversing the
3202 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfFilterCollection
3203 association adaptation (see 7.3.18) by invoking the GetAssociatedInstancesWithPath()
3204 operation with the following actual values for the input parameters:

- 3205 – InstanceName: the object path to the input IndicationService instance
- 3206 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
3207 IndicationServiceOfFilterCollection association adaptation
- 3208 – ResultClass: "CIM_FilterCollection", the adapted class of the StaticFilterCollection
3209 adaptation

3210 The result of this step is a set of StaticFilterCollection instances (see 7.3.17).

3211 2) Inspect each StaticFilterCollection instance resulting from step 1) by checking the value of the
3212 CollectionName property. If the name of the static filter collection as indicated by that value
3213 matches the desired name, this use case is complete; otherwise, continue inspecting the next
3214 IndicationFilter instance returned by step 1).

3215 8.18.3 Postconditions

3216 Unless errors occurred, the client knows the named filter collection by means of the representing
3217 StaticFilterCollection instance (including its object path).

3218 8.19 CreateSubscription: Create a subscription

3219 8.19.1 Preconditions

3220 The client knows all of the following:

- 3221 • the object path to the IndicationService instance representing the indication service within the
3222 implementation (see 7.3.2)
- 3223 • an object path to an IndicationFilter instance representing an indication filter covering the
3224 desired indication or set of indications
- 3225 For example, see the FindIndicationFilter (8.12) or CreateIndicationFilter (8.14) use cases about
3226 how to obtain that object path.
- 3227 • Alternatively, an object path to a StaticFilterCollection instance representing a filter collection
3228 covering the desired indication or set of indications. For example, see the
3229 ObtainNamedCollection use case (8.18) about how to obtain the object path to a
3230 StaticFilterCollection instance representing a global filter collection or a profile-specific filter
3231 collection.
- 3232 • an object path to a ListenerDestination instance representing a listener destination that
3233 represents the desired listener within the implementation. For example, see the
3234 SelectListenerDestination use case (8.7) about how to obtain that object path.

3235 8.19.2 Flow of activities

- 3236 1) Prepare a local instance of the CIM_IndicationSubscription class (or the
3237 CIM_FilterCollectionSubscription for a subscription to a filter collection) that complies with the
3238 requirements of the FilterSubscription adaptation (see 7.3.26) or the CollectionSubscription
3239 adaptation (see 7.3.27), inserting property values as follows:
 - 3240 – Filter: input object path to the indication filter (or to the filter collection)
 - 3241 – Handler: input object path to the listener destination
- 3242 The values of other properties should be specified in conformance with the capabilities of the
3243 implementation as exposed by instances of the IndicationService adaptation and the
3244 IndicationServiceCapabilities adaptation; see the DetermineIndicationServiceCapabilities use
3245 case (8.4) to obtain knowledge about these capabilities.
- 3246 Values not described through these adaptations may or may not be respected by the
3247 implementation; in this case it is implementation dependent whether in step 2) the
3248 implementation imposes a respective default behavior, or whether it fails in creating the new
3249 subscription.
- 3250 2) Define the new subscription to the implementation by invoking the CreateInstance() operation,
3251 providing the CIM_IndicationSubscription (or CIM_FilterCollectionSubscription) instance
3252 prepared in step 1) as the actual value of the NewInstance parameter.
- 3253 If successful, the operation returns the object path of the DynamicIndicationFilter instance
3254 representing the newly created subscription.
- 3255 If not successful, the operation returns a CIM status code providing details about the failure
3256 (see 7.3.26.3.2 or 7.3.27.3.2).

3257 8.19.3 Postconditions

- 3258 Unless errors occurred, the client knows the object path of an AbstractSubscription instance representing
3259 the newly created subscription; otherwise, the client knows details about why it was not possible to create
3260 the subscription.

3261 **8.20 CheckSubscriptions: Determine whether subscriptions exist for a given**
3262 **indication and listener**

3263 **8.20.1 Preconditions**

3264 The client knows all of the following:

- 3265 • the object path to the IndicationService instance representing the indication service within the
3266 implementation (see 8.2)
- 3267 • the URI exposed by the desired listener

3268 **8.20.2 Flow of activities**

- 3269 1) Execute the ListListenerDestinations use case (see 8.6). The result is a set of
3270 ListenerDestination instances (including their object paths) representing all the listener
3271 destinations within the implementation.
 - 3272 2) From the result of step 1), drop all ListenerDestination instances not referencing the desired
3273 listener. The result is a set of ListenerDestination instances (including their object paths)
3274 representing all the listener destinations referencing the desired listener.
 - 3275 3) For each ListenerDestination instance resulting from step 2), find all IndicationFilter instances
3276 (see 7.3.11) associated with the ListenerDestination instance (see 7.3.23) through a
3277 FilterSubscription instance (see 7.3.26). The result of this step is a set of IndicationFilter
3278 instances representing indication filters to which the desired listener is subscribed.
 - 3279 4) Inspect each IndicationFilter instance resulting from step 3) by checking the values of the
3280 QueryLanguage and the Query properties. Interpret the query statement expressed by the value
3281 of the Query property and check whether the input indication is covered. If the input indication is
3282 covered, add the identification of the represented listener destination to a filter result list, and
3283 continue inspecting the next IndicationFilter instance returned by step 3).
 - 3284 5) For each ListenerDestination instance resulting from step 2), find all StaticFilterCollection
3285 instances (see 7.3.17) associated through a CollectionSubscription instance (see 7.3.27). The
3286 result of this step is a set of StaticFilterCollection instances representing static filter collections
3287 to which the desired listener is subscribed.
 - 3288 6) For each StaticFilterCollection instance resulting from step 5), apply the
3289 CheckCollectionCoverage use case (see 8.17).
- 3290 If the input indication is covered, add the identification of the represented static filter collection to
3291 a collection result list, and continue inspecting the next StaticFilterCollection instance returned
3292 by step 5).

3293 **8.20.3 Postconditions**

3294 Unless errors occurred, the client knows (the identifications of) all listener destinations and filter
3295 collections to which the desired listener is subscribed.

3296 **8.21 DeleteSubscription: Delete a subscription**

3297 **8.21.1 Preconditions**

3298 The client knows all of the following:

- 3299 • the object path to the AbstractSubscription instance (see 7.3.25) representing a subscription
3300 within the implementation

3301 **8.21.2 Flow of activities**

3302 1) Invoke the DeleteInstance() operation on the AbstractSubscription instance, effecting the
3303 deletion of the represented subscription.

3304 NOTE If the subscription referenced a dynamic indication filter, and no other subscriptions reference it, and the
3305 client does not plan to create a new subscription for this filter, the client can delete the dynamic indication
3306 filter using the DeleteFilter use case (see 8.16); likewise, unless referenced by other subscriptions, the
3307 client can delete the listener destination that was referenced by the deleted subscription, using the
3308 DeleteListenerDestination use case (see 8.11).

3309 **8.21.3 Postconditions**

3310 Unless errors occurred, the subscription is deleted and no longer represented by any
3311 AbstractSubscription instance.

3312 **8.22 FindAlertingSystem: Find the system containing a component causing an
3313 alert indication**3314 **8.22.1 Preconditions**

3315 The client knows all of the following:

- 3316 • an AlertIndication instance representing an alert indication that references the alerting managed
3317 element

3318 **8.22.2 Flow of activities**

3319 1) Obtain the CIM element referenced by the value of the AlertingManagedElement in the input
3320 AlertIndication instance.

3321 2) Determine the profile with which the CIM element is conformant and where the central class
3322 adaption adapts the CIM_System class.

3323 NOTE This step implies client knowledge about profiles defining adaptations of the class of the CIM
3324 element obtained in step 1). More than one profile could impact the CIM element, but the
3325 scoping CIM_System instance should be the same in all cases.

3326 3) Use the scoping algorithm defined by the profile determined in step 2) to find the related
3327 instance of the scoping class adaptation of that profile.

3328 **8.22.3 Postconditions**

3329 Unless errors occurred, the client knows the CIM_System instance representing the system containing a
3330 component causing the generation of the input alert indication.

3331 **8.23 DetermineIndicationGate: Determine the indication gate of an indication**3332 **8.23.1 Preconditions**

3333 The client knows all of the following:

- 3334 • an AlertIndication instance representing an alert indication that references the alerting managed
3335 element

3336 In addition, subscriptions for the listener that received the input alert indication should have been
3337 established such that within the set of subscribed to indication gates within a particular implementation
3338 each is uniquely identified with a name as exposed by the value of the Name property in representing

3339 IndicationFilter instances (see 7.3.11), or as exposed by the value of the CollectionName property in
 3340 representing StaticFilterCollection instances (see 7.3.17).

3341 NOTE This policy ensures that indication gate names are unique with respect to one implementation;
 3342 implementations are unable to (and not required to) maintain that uniqueness, but clients can ensure it
 3343 through carefully applying the subscription policy stated above for each listener that a client controls.

3344 8.23.2 Flow of activities

3345 1) Extract the value of the IndicationFilterName from the input AlertIndication instance as the name
 3346 of the sought-after indication gate.

3347 If the input alert indication originates from an implementation that is known to the client by
 3348 reference to its representing IndicationFilter instance, skip to step 8); otherwise, continue with
 3349 step 2).

3350 2) Inspect the value of the AlertingManagedElement property of the input AlertIndication instance.

3351 If that value is Null, then the indication gate cannot be determined, and this use case is
 3352 complete without success; this is also the case of the value is a URI that does not reference a
 3353 CIM instance that represents the alerting managed element. In subsequent steps it is assumed
 3354 that the value is a URI that references a CIM instance that represents the alerting managed
 3355 element.

3356 3) Determine the ProfileRegistration instance that is providing the CIM instance referenced by the
 3357 URI found in step 2), using one of the algorithms described in [DSP1033](#) for that purpose.

3358 4) Apply the LocateProfileIndicationService use case (see 8.3) in order to determine the
 3359 IndicationService instance (see 7.3.2) that represents the indication service from which the input
 3360 alert indication originated.

3361 5) Find all IndicationFilter instances (see 7.3.11) associated with the IndicationFilter instance (see
 3362 7.3.23) found in step 4) through an IndicationServiceOfIndicationFilter instance (see 7.3.14), for
 3363 example by executing the GetAssociatedInstancesWithPath() operation.

3364 6) For each IndicationFilter instance obtained in step 5), determine if the value of the Name
 3365 property matches the name of the sought-after indication gate determined in step 1).

3366 If it matches, and the subscription policy mentioned in the preconditions was maintained, then
 3367 the indication filter represented by the IndicationFilter instance is the sought-after indication
 3368 gate.

3369 If the name matches, and the subscription policy was not maintained, then all IndicationFilter
 3370 instances determined in step 5) need to be checked with step 6) in order to ensure that the
 3371 name as exposed by the value of the Name property is not used more than once. If this is the
 3372 case, the sought-after indication gate cannot be exactly determined; however, at least it can be
 3373 limited to the set of indication filters using the name as determined in step 1).

3374 If a name does match, continue with step 8).

3375 If the name does not match, the next instance from the set determined in step 5) needs to be
 3376 checked with step 6); if no additional instances remain, continue with step 7).

3377 7) Repeat steps 5) and 6) for filter collections, searching for StaticFilterCollection instances (see
 3378 7.3.17) associated through an IndicationServiceOfFilterCollection instance (see 7.3.18) in step
 3379 5), and checking the value of the CollectionName property in step 6).

3380 8) If an indication filter was determined as the sought-after indication gate in steps 1), 6), or 7), the
 3381 client can check the query statement exposed by the value of the Query property in the
 3382 representing IndicationFilter instance (or — in case the alert indication was received through a
 3383 filter collection — in at least one of the contained IndicationFilter instances), and verify that the

3384 input alert indication is indeed within the coverage of the identified indication filter or filter
3385 collection.

3386 8.23.3 Postconditions

3387 Unless errors occurred, the client knows the indication gate emitting the input alert indication by means of
3388 its representing IndicationFilter or StaticFilterCollection instance.

3389 8.24 SubscribeForProfileIndications: Subscribe for all of the indications defined 3390 in a referencing profile

3391 8.24.1 Preconditions

3392 The client knows the following:

- 3393 • the registered name of the referencing profile
- 3394 • the object path to the IndicationService instance representing the indication service within the
3395 implementation (see 7.3.2)
- 3396 • the object path to the ListenerDestination instance (see 7.3.23) representing the desired listener
3397 destination

3398 8.24.2 Flow of activities

- 3399 1) Construct the name for the profile-specific filter collection for alert indications, applying the
3400 pattern defined in 7.3.21.2.2.
- 3401 2) Execute the ObtainNamedCollection use case (see 8.18), providing the name constructed in
3402 step 1) as input; the result is either Null or the object path referencing the
3403 ProfileSpecificAlertIndicationFilterCollection instance (see 7.3.21) representing the profile-
3404 specific filter collection for alert indications of the referencing profile.
- 3405 3) If an object path was returned on step 2), execute the CreateSubscription use case (see 8.19),
3406 providing that object path and the input object path to the ListenerDestination instance as input.
- 3407 4) Perform steps 1), 2) and 3) analogously for lifecycle indications.

3408 8.24.3 Postconditions

3409 Unless errors occurred, the desired listener destination is subscribed for all alert indications and all
3410 lifecycle indications defined by the referencing profile.

3411

ANNEX A (informative)

3412
3413
3414
3415

Profiles defining indications

3416 Referencing profiles define indications and related requirements in the following ways:

- 3417 • Reference this profile as a mandatory or conditional profile
- 3418 • Define lifecycle indications and/or alert indications by defining adaptations based on the
3419 LifecycleIndication adaptation (see 7.3.32) and/or the AlertIndication adaptation (see 7.3.31).
3420 This requires but is not limited to defining the requirement level, the reported event, and the
3421 query statement; however, the latter two may be implied by the respective base adaptation.
- 3422 • Optionally, define indication filters by defining adaptations based on the StaticIndicationFilter
3423 adaptation (see 7.3.11). The definition of indication-specific indication filters covering each
3424 lifecycle indication and each alert indication defined in a referencing profile is implied by this
3425 profile through the IndicationSpecificIndicationFilter adaptation (see 7.3.15), but may be refined
3426 by referencing profiles.
- 3427 • Optionally, define filter collections by defining adaptations based on the StaticFilterCollection
3428 adaptation (see 7.3.17). The definition of profile-specific filter collections covering all lifecycle
3429 indications and/or alert indications defined in a referencing profile is implied by this profile
3430 through the ProfileSpecificFilterCollection adaptation (see 7.3.21), but may be refined by
3431 referencing profiles.

**ANNEX B
(informative)**

Change Log

3432
3433
3434
3435

Version	Date	Description
1.0.0a	2007-06-04	Preliminary Standard
1.0.0	2008-12-05	Final Standard
1.0.1	2009-09-07	Released as DMTF Standard, with the following changes: <ul style="list-style-type: none"> • Updated profile conventions for operations and their usage • Fixed incorrect CIM Schema version (from 2.16 to 2.22)
1.1.0a	2009-12-02	Released as Work in Progress, with the following changes: <ul style="list-style-type: none"> • Increased CIM Schema version to 2.23(exp). • Added support for reliable indications (delivery retry, detection of lost indications, reconstruction of original order): <ul style="list-style-type: none"> – Description of reliable indications concept in 7.10 (Indication Delivery). – Clarifications in description of CIM_ListenerDestination.PersistenceType. • Refined the format for CIM_FilterCollection.CollectionName in 7.6. • Refined the format for CIM_IndicationFilter.Name in 7.4. • Cleaned up terminology clause by removing most terms that are defined in DSP0004, DSP0200 or DSP1001. • Added "Document conventions" clause and consolidated existing text into that. • Updated profile conventions for operations to match DSP1001 1.0.1. • Fixed incorrect pattern value "WBEMURI" for CIM_AlertIndication.AlertingElementFormat.
1.1.0	2010-05-20	Released as DMTF Standard, with the following changes: <ul style="list-style-type: none"> • Clarified and added some terms in clause 3. • Clarified that there is only one indication service in a WBEM server, but added a recommendation for clients to expect more than one in the future. • Fixed incorrect verbiage of sending indications to clients, to sending indications to listeners. • Changed ambiguous "conditional/optional" requirement to "conditional or optional" in all cases but one. • Clarified that listeners that intend to re-establish the original order of indications need to buffer indications that do not have the predicted sequence number until decision about loss can be made. • Lowered the requirement not to interpret sequence numbers in case of not implementing them, to a permission to ignore them. • Fixed inconsistencies in several diagrams.

Version	Date	Description
1.2.0a	2010-06-16	<p>Released as Work in Progress, with the following changes:</p> <ul style="list-style-type: none"> • Increased CIM Schema version to 2.25 • Converted to PUG 1.1 "Condensed Format": <ul style="list-style-type: none"> – The semantics of the definitions from the previous version was maintained, except the semantical changes detailed – Introduced separation between managed environment and CIM model – Defined many new terms precisely capturing concepts only vaguely defined in the previous version – Introduced features – Introduced adaptations, integrating the content of the Methods and the "CIM elements" clauses defined in the previous version into the "Implementation" clause of this version – Modified existing use cases using adaptations, and introduced new use cases • Introduced the following new concepts: <ul style="list-style-type: none"> – Global filters – Global filter collections – Profile-specific filters – Profile-specific filter collections • Deprecated the use of the CIM_ConcreteDependency association for modeling the relationship between filter collections and profile representations of referencing profiles (CIM_RegisteredProfile) • Changed the requirement level for the ElementConformsToProfile association adaptation to mandatory • Fixed incorrect property name: CIM_ListenerDestination.Protocol was incorrectly named ProtocolType. • Many clarifications of existing concepts, such as the following: <ul style="list-style-type: none"> – Established indication emitters as the super type of indication filters and filter collections – Clarified that the purpose of indication emitters is filtering indications, and is not the representation of indication implementations – Restructured the specification of reliable indications using a feature and adaptations – Consistently use the terms "sequence identifier" and "sequence identifier lifetime", as established by the CIM schema (quit using the term "sequence identifier value") – Suppression of repeated indication delivery

Version	Date	Description
1.2.0b	2010-09-15	<p>Released as Work in Progress, with the following changes:</p> <ul style="list-style-type: none"> • Included cPubs major scrub • Renamed indication emitter -> indication gate • Renamed profile-specific filter -> indication-specific filter • Removed specializations of these (introduced in the 1.2.0a version) • Deprecate requiring a CIM_Error instance in case of IndicationFilter.CreateInstance() error • Recommending the Interop namespace as the only namespace for IndicationFilter instances, StaticFilterCollection instances, ListenerDestination instances and AbstractSubscription instances • Many clarifications of existing concepts and addition of new concepts, such as the following: <ul style="list-style-type: none"> – Require that all kinds of filters have one or more related namespaces, either those identified by the value of the SourceNamespaces[] property, or – if the value is Null - the namespace where the filter representation resides; version 1.1 left that open for dynamic filters. – Prohibit empty array as possible SourceNamespaces[] value, as that would be semantically useless because no indication would be allowed to pass in this case. – Requiring that indications have an origin namespace – Requiring that the origin namespace is taken into consideration during indication filtering, i.e., is subject to filtering. This is done by extending the concept of the filter coverage such that both query statement and the namespace list span the filter coverage – Correcting the prohibition of providing any key properties when creating dynamic indication filters by exempting the Name property, along with a recommended naming convention

Version	Date	Description
1.2.0c	2011-04-05	<p>Released as a DMTF Draft Standard, with the following changes:</p> <ul style="list-style-type: none"> • Adjusted to the DMTF Draft Standard version of DSP1001 1.1 <ul style="list-style-type: none"> – Moved base elements from the table of class adaptations to the individual element requirements tables of the adaptations – Adopted the format for error reporting requirements – Require Key properties to be listed when used the first time in a chain of adaptations – Introduction of the implementation type – Restructured the error reporting requirement tables • Minor corrections resulting from reviews of version 1.2.0b • Adjust the use of the Profile Registration profile to DSP1033 1.0 • Specify operation requirements in terms of DSP0223 (as required by DSP1001, after Architecture workgroup decision) • Rephrased the policies for the avoidance of repeated indication delivery, synchronizing it with the phraseology used in the schema description of the CIM_AbstractIndicationSubscription class • Resolved various comments from 2nd SNIA review • Changed the requirement level of IndicationServiceOfIndicationFilter, IndicationServiceOfFilterCollection, CollectionSubscription and ProfileOfFilterCollection from conditional to mandatory because the condition was always true (the GlobalFilter and GlobalFilterCollection adaptations are mandatory derived adaptation of the IndicationFilter and StaticFilterCollection adaptations) • Reinforced the version 1.1 requirement that key properties on the creation of DynamicIndicationFilter instances are to be ignored, and should not be provided by clients • Extended the AlertIndication adaptation to allow for referencing more than one alert message • Extended the IndicationSpecificIndicationFilter adaptation to provide for multiple instances for the coverage of multi-message AlertIndication adaptations
1.2.0	2011-06-30	<p>Released as a DMTF Standard, with the following changes:</p> <ul style="list-style-type: none"> • Confirmed the CIM schema definition of CIM_Indication wrt. that a sequence identifier needs to be maintained on a per listener destination basis (and not on a per listener basis)
1.2.1	2011-10-26	<p>Released as a DMTF Standard, with the following errata corrected:</p> <ul style="list-style-type: none"> • Allow OrgID values other than "DMTF" as first part of the value of the InstanceID property in ProfileSpecificFilterCollection instances • Fix copy/paste error in GlobalFilter element requirement table • Fix value constraint for the IndicationFilter.QueryLanguage property to "DMTF:CQL" • Updated owning working group (Architecture) and author list.