



1
2
3
4
5

Document Number: DSP1054

Date: 2014-04-24

Version: 1.2.2

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, 2014 Distributed Management Task Force, Inc. (DMTF). All rights
13 reserved.

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

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28 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

CONTENTS

35	Foreword	7
36	Introduction.....	8
37	Document conventions.....	8
38	Typographical conventions	8
39	ABNF usage conventions	8
40	Deprecated material.....	8
41	Experimental material	8
42	1 Scope	11
43	2 Normative references	11
44	3 Terms and definitions	12
45	4 Symbols and abbreviated terms.....	16
46	5 Synopsis	16
47	6 Description	19
48	6.1 Events and indications	19
49	6.1.1 Events	19
50	6.1.2 Indications.....	19
51	6.1.3 Definition of events and indications in referencing profiles.....	21
52	6.1.4 Indication generation, indication filtering, and indication delivery.....	21
53	6.1.5 Reliable indication delivery	23
54	6.1.6 Avoidance of repeated indication delivery	23
55	6.2 Indication filters	24
56	6.2.1 General	24
57	6.2.2 Indication filter coverage.....	25
58	6.2.3 Static indication filters	26
59	6.2.4 Indication-specific indication filters	26
60	6.2.5 Global indication filters.....	27
61	6.2.6 Dynamic indication filters	27
62	6.3 Filter collections	27
63	6.3.1 General	27
64	6.3.2 Filter collection coverage	27
65	6.3.3 Static filter collections	28
66	6.4 Subscriptions, listeners, and listener destinations.....	28
67	6.4.1 Subscriptions	28
68	6.4.2 Overlapping coverages of subscriptions.....	29
69	6.4.3 Subscription management authorization	29
70	6.4.4 Listeners	29
71	6.4.5 Listener destinations	29
72	6.5 Indication service and implementation.....	30
73	6.5.1 Implementation	30
74	6.5.2 Indication service	30
75	6.6 Indication system and referencing profiles	30
76	6.7 CIM model.....	31
77	7 Implementation.....	34
78	7.1 Separation of requirements	34
79	7.2 Features.....	34
80	7.2.1 DynamicIndicationFilters.....	34
81	7.2.2 IndicationServiceInitialSettingsExposed	35
82	7.2.3 IndicationServiceModification	35
83	7.2.4 ReliableIndications.....	35
84	7.2.5 SuppressRepeatNotificationPolicy.....	35
85	7.2.6 DelayRepeatNotificationPolicy.....	36

86	7.2.7	IndividualFilterSubscription	36
87	7.2.8	FilterCollectionCoverageExposure	36
88	7.2.9	LifeCycleGlobalIndicationFilter	37
89	7.2.10	AlertGlobalIndicationFilter	37
90	7.3	Adaptations	37
91	7.3.1	Conventions	37
92	7.3.2	IndicationService: CIM_IndicationService	38
93	7.3.3	IndicationSystem: CIM_System	41
94	7.3.4	HostedIndicationService: CIM_HostedService	42
95	7.3.5	IndicationsProfileRegistration: CIM_RegisteredProfile	42
96	7.3.6	ElementConformsToProfile: CIM_ElementConformsToProfile	43
97	7.3.7	IndicationServiceCapabilities: CIM_IndicationServiceCapabilities	44
98	7.3.8	CapabilitiesOfIndicationService: CIM_ElementCapabilities	45
99	7.3.9	IndicationServiceInitialSettings: CIM_IndicationServiceSettingData	46
100	7.3.10	InitialSettingsOfIndicationService: CIM_ElementSettingData	47
101	7.3.11	IndicationFilter: CIM_IndicationFilter	48
102	7.3.12	StaticIndicationFilter: CIM_IndicationFilter	51
103	7.3.13	DynamicIndicationFilter: CIM_IndicationFilter	51
104	7.3.14	IndicationServiceOfIndicationFilter: CIM_ServiceAffectsElement	55
105	7.3.15	IndicationSpecificIndicationFilter: CIM_IndicationFilter	55
106	7.3.16	GlobalIndicationFilter: CIM_IndicationFilter	57
107	7.3.17	StaticFilterCollection: CIM_FilterCollection	58
108	7.3.18	IndicationServiceOfFilterCollection: CIM_OwningCollectionElement	61
109	7.3.19	IndicationFilterInFilterCollection: CIM_MemberOfCollection	62
110	7.3.20	FilterCollectionInFilterCollection: CIM_MemberOfCollection	62
111	7.3.21	ProfileSpecificFilterCollection: CIM_FilterCollection	63
112	7.3.22	GlobalFilterCollection: CIM_FilterCollection	65
113	7.3.23	ListenerDestination: CIM_ListenerDestination	67
114	7.3.24	IndicationServiceOfListenerDestination: CIM_ServiceAffectsElement	71
115	7.3.25	AbstractSubscription: CIM_AbstractIndicationSubscription	71
116	7.3.26	FilterSubscription: CIM_IndicationSubscription	75
117	7.3.27	CollectionSubscription: CIM_FilterCollectionSubscription	77
118	7.3.28	ProfileOfFilterCollection: CIM_ConcreteDependency	79
119	7.3.29	BasicIndication: CIM_Indication	79
120	7.3.30	ReliableIndication: CIM_Indication	82
121	7.3.31	AlertIndication: CIM_AlertIndication	83
122	7.3.32	LifecycleIndication: CIM_InstIndication	87
123	7.3.33	ListenerDestinationRemovalIndication: CIM_InstDeletion	89
124	7.3.34	SubscriptionRemovalIndication: CIM_InstDeletion	89
125	7.4	Reliable indication delivery	90
126	7.4.1	General	90
127	7.4.2	Sequence identifier and sequence identifier lifetime	90
128	7.4.3	WBEM server requirements	91
129	7.4.4	WBEM listener requirements	92
130	8	Use cases	94
131	8.1	Object diagrams	94
132	8.2	LocateIndicationService: Locate the indication service provided by an implementation of this profile	98
133	8.3	LocateProfileIndicationService: Locate the indication service responsible for delivering indications defined by a referencing profile	99
134	8.4	DetermineIndicationServiceCapabilities: Determine the capabilities of an indication service	99
135	8.5	ModifyIndicationService: Modify functional aspects of an indication service	101
136	8.6	ListListenerDestinations: List all listener destinations exposed by an implementation	101
137	8.7	SelectListenerDestination: Select an existing listener destination referencing a desired listener	102
138			
139			
140			
141			

142 8.8 CreateListenerDestination: Create a new listener destination 102

143 8.9 FindFreeListenerDestination: Find a free listener destination 103

144 8.10 ModifyListenerDestination: Modify an existing listener destination 104

145 8.11 DeleteListenerDestination: Delete an existing listener destination..... 104

146 8.12 FindIndicationFilter: Find an indication filter covering a particular indication 105

147 8.13 DetermineQueryLanguages: Determine the set of query languages supported for query

148 statements 105

149 8.14 CreateIndicationFilter: Create a dynamic indication filter covering a particular indication 106

150 8.15 ModifyIndicationFilter: Modify a dynamic indication filter..... 107

151 8.16 DeleteIndicationFilter: Delete a dynamic indication filter 107

152 8.17 CheckCollectionCoverage: Check the coverage of a filter collection 108

153 8.18 ObtainNamedCollection: Obtain a named filter collection 109

154 8.19 CreateSubscription: Create a subscription 110

155 8.20 CheckSubscriptions: Determine whether subscriptions exist for a given indication and

156 listener 111

157 8.21 DeleteSubscription: Delete a subscription 112

158 8.22 FindAlertingSystem: Find the system containing a component causing an alert indication... 112

159 8.23 DetermineIndicationGate: Determine the indication gate of an indication..... 113

160 8.24 SubscribeForProfileIndications: Subscribe for all of the indications defined in a

161 referencing profile 114

162 ANNEX A (informative) Profiles defining indications 115

163 ANNEX B (informative) Change log..... 116

Figures

166 Figure 1 – Indication related functionality within an implementation 22

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

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

169 Figure 4 – DMTF object diagram: Global and profile-specific filter collections..... 94

170 Figure 5 – DMTF object diagram: Filter collections and contained indication filters..... 96

171 Figure 6 – DMTF object diagram: Static listener destinations 97

172

Tables

174 Table 1 – Profile references 17

175 Table 2 – Adaptations 17

176 Table 3 – Features 18

177 Table 4 – IndicationService: Element requirements 38

178 Table 5 – ModifyInstance(): Error reporting requirements 40

179 Table 6 – IndicationSystem: Element requirements 41

180 Table 7 – HostedIndicationService: Element requirements 42

181 Table 8 – IndicationsProfileRegistration: Element requirements 43

182 Table 9 – ElementConformsToProfile: Element requirements 43

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

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

185 Table 12 – IndicationServiceInitialSettings: Element requirements 46

186 Table 13 – InitialSettingsOfIndicationService: Element requirements 47

187 Table 14 – IndicationFilter: Element requirements 49

188 Table 15 – StaticIndicationFilter: Element requirements 51

189 Table 16 – DynamicIndicationFilter: Element requirements 52

190 Table 17 – CreateInstance(): Error reporting requirements 52

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

192 Table 19 – ModifyInstance(): Error reporting requirements 54

193 Table 20 – IndicationServiceOfIndicationFilter: Element requirements 55

194 Table 21 – IndicationSpecificIndicationFilter: Element requirements 56

195 Table 22 – GlobalIndicationFilter: Element requirements 57

196 Table 23 – GlobalIndicationFilter: Instance requirements for instances covering all alert indications 58

197 Table 24 – GlobalIndicationFilter: Instance requirements for instances covering all lifecycle indications.. 58

198 Table 25 – StaticFilterCollection: Element requirements 60

199 Table 26 – IndicationServiceOfFilterCollection: Element requirements 61

200 Table 27 – IndicationFilterInFilterCollection: Element requirements 62

201 Table 28 – FilterCollectionInFilterCollection: Element requirements 63

202 Table 29 – ProfileSpecificFilterCollection: Element requirements 63

203 Table 30 – GlobalFilterCollection: Element requirements 65

204 Table 31 – ListenerDestination Element requirements 67

205 Table 32 – CreateInstance(): Error reporting requirements 69

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

207 Table 34 – ModifyInstance(): Error reporting requirements 70

208 Table 35 – IndicationServiceOfListenerDestination: Element requirements 71

209 Table 36 – AbstractSubscription: Element requirements 72

210 Table 37 – RepeatNotificationPolicy: Value constraints 73

211 Table 38 – ModifyInstance(): Error reporting requirements 75

212 Table 39 – FilterSubscription: Element requirements 76

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

214 Table 41 – CollectionSubscription: Element requirements 77

215 Table 42 – CreateInstance(): Error reporting requirements 78

216 Table 43 – ProfileOfFilterCollection: Element requirements 79

217 Table 44 – BasicIndication: Element requirements 81

218 Table 45 – ReliableIndication: Element requirements 82

219 Table 46 – AlertIndication: Element requirements 85

220 Table 47 – LifecycleIndication: Element requirements 88

221 Table 48 – ListenerDestinationRemovalIndication: Element requirements 89

222 Table 49 – SubscriptionRemovalIndication: Element requirements 89

223

224

Foreword

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

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

230 Acknowledgments

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

232 Editor:

- 233 • Michael Johanssen, IBM

234 Contributors:

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

244

Introduction

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

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

251 Document conventions

252 Typographical conventions

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

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

259 ABNF usage conventions

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

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

264 Deprecated material

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

270 The following typographical convention indicates deprecated material:

271 DEPRECATED

272 Deprecated material appears here.

273 DEPRECATED

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

276 Experimental material

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

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

282 The following typographical convention indicates experimental material:

283 **EXPERIMENTAL**

284 Experimental material appears here.

285 **EXPERIMENTAL**

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

288

290

Indications Profile

291 1 Scope

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

295 2 Normative references

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

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

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

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

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

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

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

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

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

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

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

320 **3 Terms and definitions**

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

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

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

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

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

337 **3.1**

338 **alert indication**

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

341 **3.2**

342 **client**

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

345 **3.3**

346 **coverage**

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

349 **3.4**

350 **defined coverage**

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

353 **3.5**

354 **dynamic indication filter**

355 an indication filter whose lifecycle is controlled by a client

356 **3.6**

357 **event**

358 an observable occurrence of a phenomenon of interest
359 For details, see 6.1.

- 360 **3.7**
361 **filter collection**
362 an indication gate that may contain other indication gates such as indication filters or other filter
363 collections
364 For details, see 6.3.
- 365 **3.8**
366 **global indication filter**
367 an indication filter that covers large sets of indications, such as all alert indications
368 For details, see 6.2.5.
- 369 **3.9**
370 **global filter collection**
371 a filter collection that covers large sets of indications, such as all lifecycle indications
372 For details, see 6.3.3.5.
- 373 **3.10**
374 **implementation**
375 a WBEM server that implements applicable portions of this profile and of referencing profiles
376 For details, see [DSP1001](#).
- 377 **3.11**
378 **indication**
379 the notification about an event that occurred
380 For details, see 6.1.
- 381 **3.12**
382 **indication delivery**
383 the process of delivering indications from an implementation to a listener
384
- 385 **indication filter**
386 an indication gate whose coverage is defined through a query statement
387 For details, see 6.2
- 388 **3.13**
389 **indication filtering**
390 the process of selecting indications based on filtering rules applied by indication gates, such that only
391 indications within the coverage of the indication gate pass the indication gate
- 392 **3.14**
393 **indication gate**
394 a managed element that filters indications such that only indications within its coverage pass. Indication
395 gates can serve as targets for subscriptions, and control which indications are delivered to subscribed
396 listeners.
- 397 **3.15**
398 **indication generation**
399 the process of creating an indication as the event that the indication is designed to report occurs
- 400 **3.16**
401 **indication origin**
402 the namespace out of that the indication originates
403 For details, see 6.1.2.4.

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

- 444 **3.27**
445 **reliable indication**
446 an indication containing a sequence identifier enabling listeners to detect duplicate, missing, or out-of-
447 order indications
448 For details, see 6.1.5 and 7.4.
- 449 **3.28**
450 **repeated indication**
451 an indication that reports the same event as a previous indication
452 For details, see 6.1.6.
- 453 **3.29**
454 **repeated indication delivery**
455 the delivery of repeated indications
456 Repeated indication delivery typically occurs if the reported event describes a persistent situation such as
457 exceeding a threshold value.
- 458 **3.30**
459 **sequence identifier**
460 data element with a reliable indication that ensures unique identification of the reliable indication
461 A sequence identifier is composed of a sequence context and a sequence number
462 For details, see 7.4.2.
- 463 **3.31**
464 **sequence identifier lifetime**
465 a maximum time interval maintained by an implementation implementing reliable indications within which
466 the implementation retries failed indication delivery attempts
467 For details, see 7.4.2.
- 468 **3.32**
469 **static filter collection**
470 a filter collection whose lifecycle is controlled by the implementation, that is uniquely identifiable and for
471 which a defined coverage is established
472 For details, see 6.3.3.
- 473 **3.33**
474 **static indication filter**
475 an indication filter whose lifecycle is controlled by the implementation
- 476 **3.34**
477 **subscription**
478 the mechanism whereby a client registers a listener for the delivery of indications from an implementation
- 479 **3.35**
480 **this profile**
481 a short term for the Indications profile, the profile specified in this specification document (DSP1054)
- 482 **3.36**
483 **WBEM client**
484 a CIM client (see [DSP0004](#)) that supports a WBEM protocol
485 For details, see [DSP1001](#).

486 **3.37**
487 **WBEM listener**
488 a CIM listener (see [DSP0004](#)) that supports a WBEM protocol
489 For details, see [DSP1001](#).

490 **3.38**
491 **WBEM server**
492 a CIM server (see [DSP0004](#)) that supports a WBEM protocol
493 For details, see [DSP1001](#).

494 **4 Symbols and abbreviated terms**

495 **4.1**
496 **CQL**
497 CIM Query Language

498 **4.2**
499 **QoS**
500 Quality of service

501 **4.3**
502 **URI**
503 Uniform Resource Identifier

504 **4.4**
505 **WBEM**
506 Web Based Enterprise Management

507 **5 Synopsis**

508 **Profile name:** Indications

509 **Version:** 1.2.2

510 **Organization:** DMTF

511 **Profile type:** Component

512 **Schema version:** 2.25

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

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

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

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

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

521 **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.

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

523 **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	Conditional	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.

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

525 **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.
LifeCycleGlobalIndicationFilter	Profile implementation	Optional	See 7.2.9.
AlertGlobalIndicationFilter	Profile implementation	Optional	See 7.2.10.

526 **6 Description**

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

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

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

538 **6.1 Events and indications**

539 **6.1.1 Events**

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

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

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

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

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

549 **6.1.2 Indications**

550 **6.1.2.1 General**

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

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

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

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

563 **6.1.2.2 Alert indications**

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

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

568 6.1.2.3 Lifecycle indications

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

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

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

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

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

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

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

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

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

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

619 **6.1.2.4 Origin of indications**

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

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

625 **6.1.3 Definition of events and indications in referencing profiles**

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

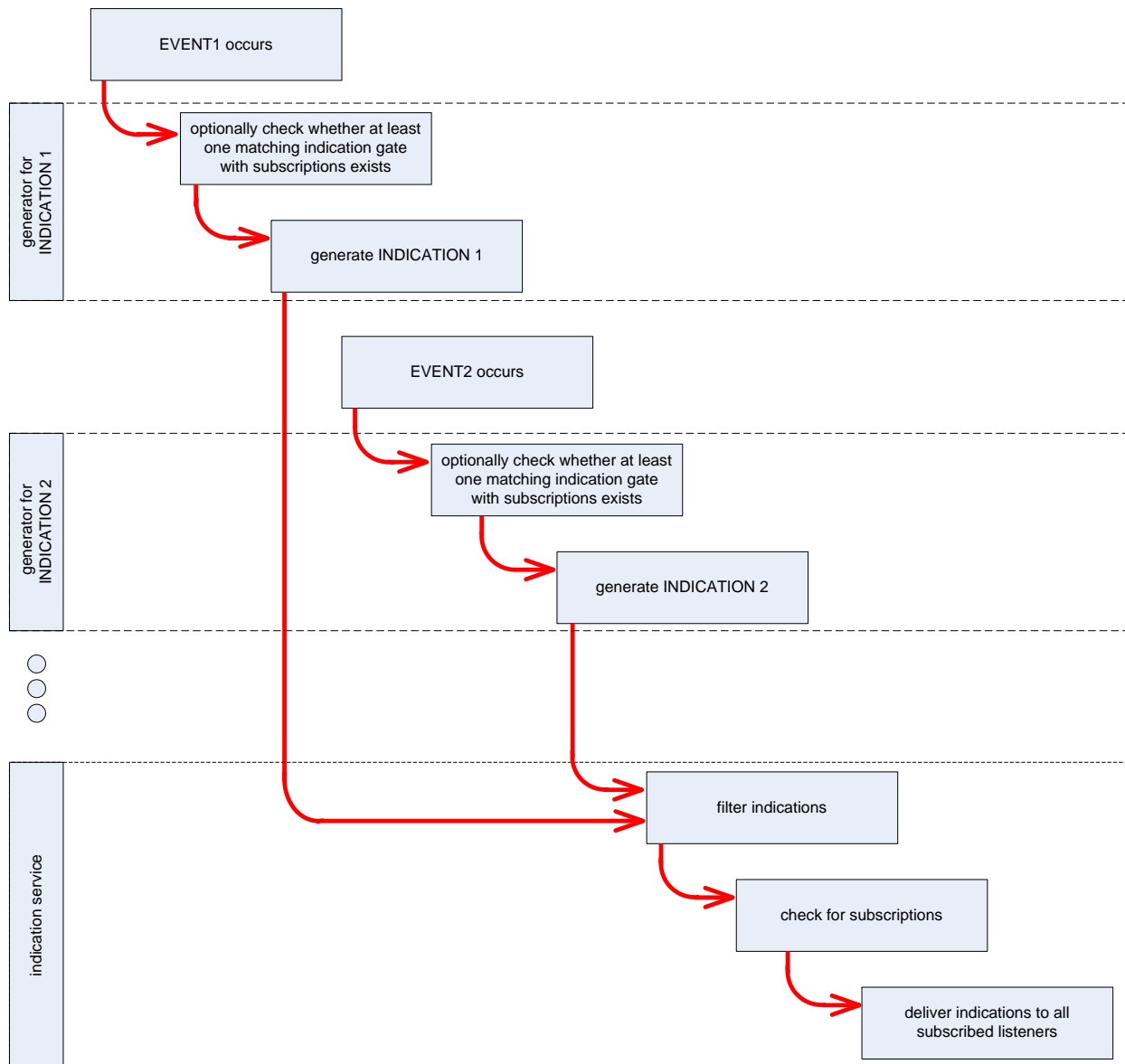
628 NOTE Defining events separately is particularly useful if multiple indications reporting the same event are
629 defined. However, if an event is only reported through one indication, the event definition as part of the
630 definition of the indication adaptation is more compact.

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

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

642 **6.1.4 Indication generation, indication filtering, and indication delivery**

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



645

646

Figure 1 – Indication related functionality within an implementation

647

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

648

649

650

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

651

652

653

654

655

656

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

657

658

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

661 Indication delivery is the process of delivering filtered indications from an implementation to a listener.
662 This profile defines rules for the delivery of indications as part of adaptations modeling indications
663 themselves, as part of adaptations modeling indication gates such as indication filters or filter collections,
664 and as part of adaptations modeling subscriptions and listener destinations. For details, see 7.3.23.2 and
665 7.3.25.2.

666 **6.1.5 Reliable indication delivery**

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

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

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

673 **6.1.6 Avoidance of repeated indication delivery**

674 **6.1.6.1 General**

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

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

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

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

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

701 **6.1.6.2 No repeated indication delivery avoidance policy**

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

705 **6.1.6.3 Suppress repeated indication delivery avoidance policy**

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

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

715 **6.1.6.4 Delayed indication delivery avoidance policy**

716 This policy is modeled by the DelayRepeatNotificationPolicy feature (see 7.2.6, and the CIM schema
717 description of the value 4 (Delay) for the RepeatNotificationPolicy property of the
718 CIM_AbstractIndicationSubscription class).

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

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

728 **6.2 Indication filters**

729 **6.2.1 General**

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

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

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

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

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

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

751 6.2.2 Indication filter coverage

752 The coverage of an indication filter is the set of indications that can pass the indication filter; it is specified
753 through an indication filter query statement and a set of namespaces identifications that identify the
754 namespaces out of which indications are filtered. In other words, only indications that originate (see
755 6.1.2.4) in one of the identified namespaces, and match the query statement pass the indication filter. For
756 details, see 7.3.11.2.

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

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

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

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

766 EXAMPLE 1:

```
767     SELECT * FROM CIM_AlertIndication
```

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

772 EXAMPLE 2:

```
773     SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA  
774     CIM_StorageVolume
```

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

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

786 **EXAMPLE 3:**

```
787 SELECT * FROM CIM_AlertIndication WHERE OwningEntity = 'DMTF' AND
788 MessageID = 'SVPC0123'
```

789 This indication filter query statement covers one alert indication. The related event is defined by
790 an alert message defined in a message repository. The value of the `OwningEntity` property
791 identifies DMTF as the organization owning the message registry. The value of the `MessageID`
792 property allows identifying the alert message within the owning organization; for details, see
793 7.3.31.

794 **EXAMPLE 4:**

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

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

799 **6.2.3 Static indication filters**

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

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

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

- 808 • Indication filters that cover more than one indication

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

- 811 • Indication filters that cover exactly one indication

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

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

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

823 **6.2.4 Indication-specific indication filters**

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

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

830 **6.2.5 Global indication filters**

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

- 834 • All alert indications
- 835 • All lifecycle indications reporting the creation of a CIM instance
- 836 • All lifecycle indications reporting the modification of a CIM instance
- 837 • All lifecycle indications reporting the deletion of a CIM instance

838 For details, see 7.3.16.

839 **6.2.6 Dynamic indication filters**

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

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

851 **6.3 Filter collections**

852 **6.3.1 General**

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

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

857 The main purposes of filter collections are:

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

865 **6.3.2 Filter collection coverage**

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

868 **6.3.3 Static filter collections**

869 **6.3.3.1 General**

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

872 **6.3.3.2 Unique identification**

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

876 **6.3.3.3 Defined coverage**

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

884 **6.3.3.4 Profile specific filter collections**

885 Profile-specific filter collection address the needs of clients requiring notifications about events reported
886 by the indications specified in a particular profile. Profile specific filter collections are a specialization of
887 static filter collections. The defined coverage of a profile-specific filter collection covers all indications of a
888 particular type (that is, all alert indications or all lifecycle indications) defined in a profile. For details, see
889 7.3.21.

890 **6.3.3.5 Global filter collections**

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

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

- 894 • All alert indications
- 895 • All alert indications specified in profiles
- 896 • All lifecycle indications
- 897 • All indications specified in profiles
- 898 • All alert indications specified in profiles
- 899 • All lifecycle indications specified in profiles

900 For details, see 7.3.22.

901 **6.4 Subscriptions, listeners, and listener destinations**

902 **6.4.1 Subscriptions**

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

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

- 906 1) Determine if there is an existing indication gate covering the desired indication set. If an
907 appropriate indication gate does not exist, and the support for dynamic indication filters is
908 implemented, the client could create dynamic indication filters (see 6.2.6).
- 909 2) Determine if a listener destination referencing the listener already exists within the
910 implementation. If such a listener destination does not yet exist, and the support for creating or
911 modifying listener destinations is implemented, the client could create a new listener destination
912 or modify an existing listener destination.
- 913 3) Create a subscription that relates the listener destination with the indication gate.

914 After it is created, a subscription results in indications being delivered to the listener that is referenced by
915 the listener destination for each event reported through any of the indications covered by the indication
916 gate referenced by the subscription.

917 **6.4.2 Overlapping coverages of subscriptions**

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

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

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

928 **6.4.3 Subscription management authorization**

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

933 **6.4.4 Listeners**

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

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

942 **6.4.5 Listener destinations**

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

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

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

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

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

958 **6.5 Indication service and implementation**

959 **6.5.1 Implementation**

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

963 **6.5.2 Indication service**

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

968 **6.6 Indication system and referencing profiles**

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

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

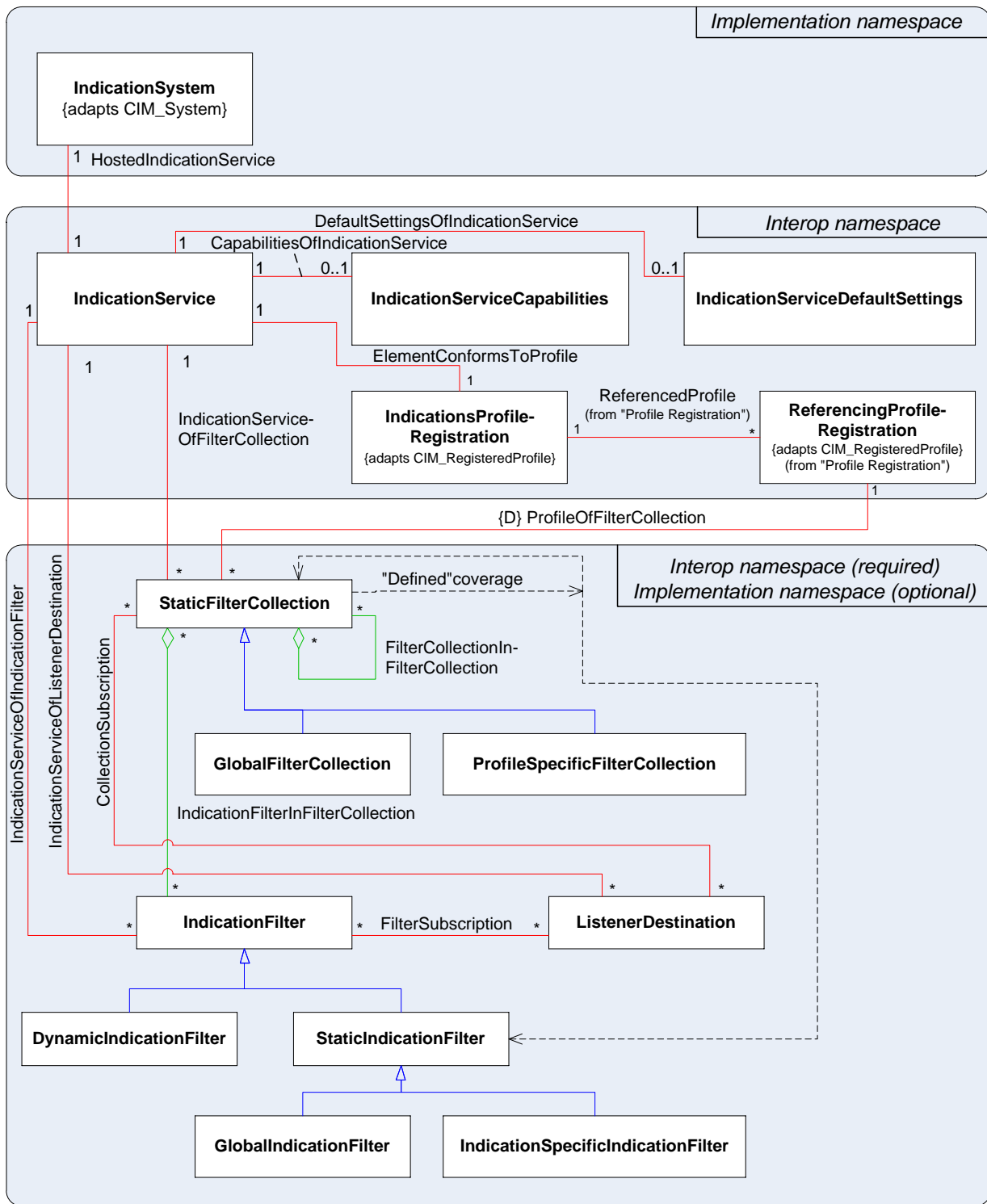
972 In the general case, the scoping systems of referencing profiles are different from the indication system,
973 that is, they are different from the system hosting the WBEM server. In other words, referencing profiles
974 are not required to provide the scope for the indication service, and the central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

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

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

986 **6.7 CIM model**

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



988

989

Figure 2 – Indications Profile: DMTF class adaptation diagram

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

- 992 • the IndicationService adaptation (see 7.3.2) models indication services as described in 6.5.2
- 993 • the IndicationFilter adaptation (see 7.3.11) models indication filters as described in 6.2
- 994 • the StaticFilterCollection adaptation (see 7.3.17) models static filter collections as described
995 in 6.3
- 996 • the StaticIndicationFilter adaptation (see 7.3.17) models static indication filters as described
997 in 6.2.3
- 998 • the ListenerDestination adaptation (see 7.3.23) models listener destinations as described
999 in 6.4.3
- 1000 • the AbstractSubscription adaptation (see 7.3.25) models subscriptions as described in 6.4.1

1001 Instances of most of these adaptations are instantiated in the Interop namespace; the use of the Interop
1002 namespace (see [DSP1033](#)) makes it easier for clients to detect the CIM representations of respective
1003 managed objects.

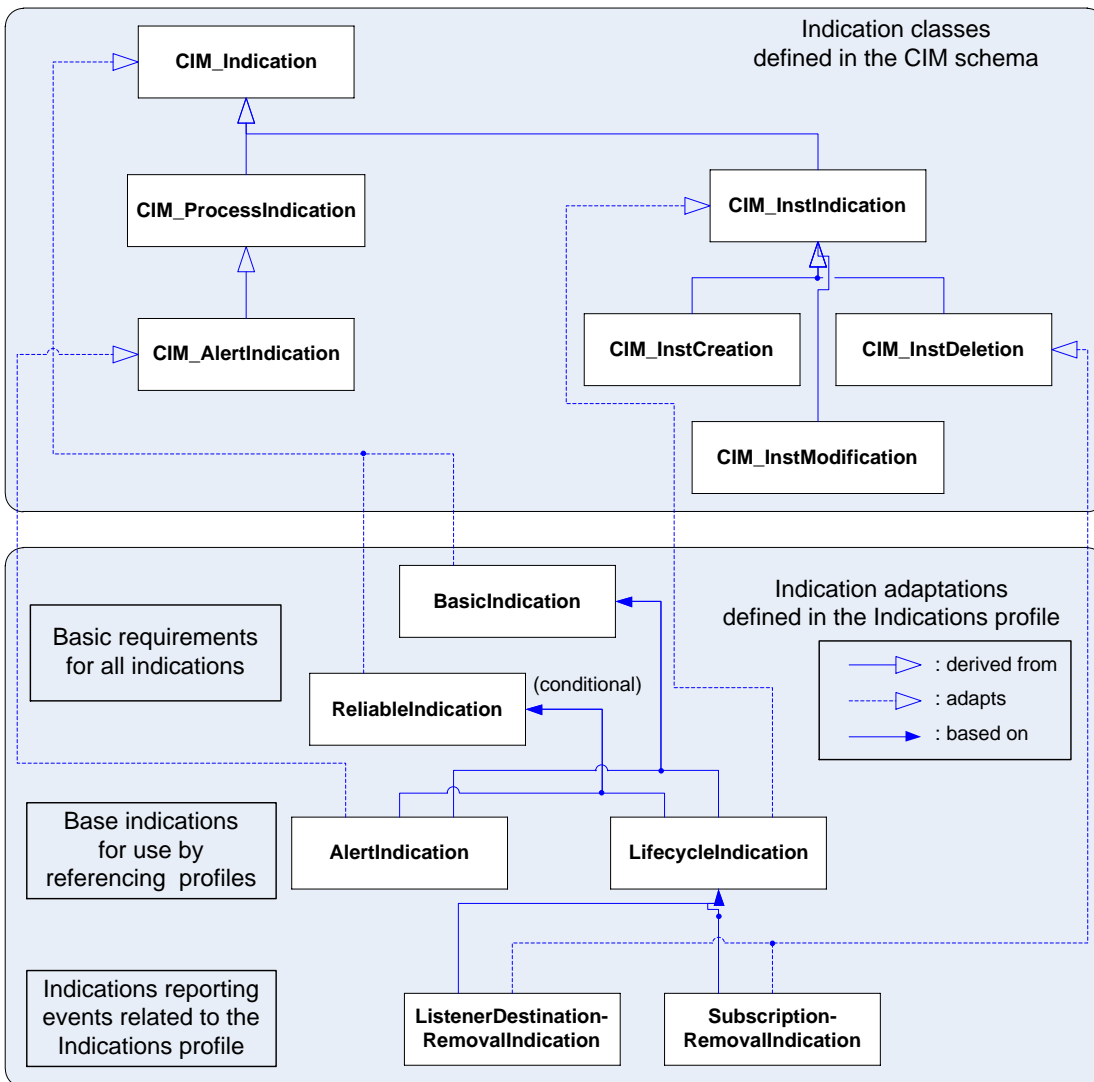
1004 **DEPRECATED**

1005 The ProfileOfFilterCollection association adaptation models the relationship between filter collections and
1006 the registration of this profile.

1007 NOTE The ProfileOfFilterCollection association adaptation (defined as the CIM_ConcreteDependency "profile
1008 class" in version 1.1 of this profile) is deprecated in version 1.2 of this profile in favor of a naming
1009 convention for static filter collections that enables their unique identification.

1010 **DEPRECATED**

1011 Figure 3 depicts the adaptations of indication classes defined by this profile along with the adapted
 1012 indication classes.



1013

1014 **Figure 3 – Indications Profile: Indication adaptations and adapted indication classes**

1015 The most essential indication adaptations defined in this profile are listed below, along with their modeled
 1016 indications:

- 1017 • the BasicIndication adaptation (see 7.3.29) models indications as described in 6.1.2
- 1018 • the ReliableIndication adaptation (see 7.3.30) models reliable indications as described in 6.1.5;
 1019 this adaptation specifies additional optional requirements that can be implemented separately
 1020 from the requirements of other indication adaptations.
- 1021 • the AlertIndication adaptation (see 7.3.31) models alert indication as described in 6.1.2.2; it is
 1022 an abstract adaptation available to referencing profiles in order to define their own alert
 1023 indications
- 1024 • the LifecycleIndication adaptation (see 7.3.32) models lifecycle indications as described
 1025 in 6.1.2.3; it is an abstract adaptation available to referencing profiles in order to define their
 1026 own lifecycle indications.

1027 7 Implementation

1028 7.1 Separation of requirements

1029 This profile defines implementation requirements for implementations (for example, WBEM servers
1030 implementing this profile) and for listeners (for example, WBEM listeners implementing this profile).

1031 The implementation requirements for implementations are further separated into WBEM server related
1032 requirements and referencing profile related requirements, as follows:

- 1033 • Requirements that address the infrastructure for the delivery of indications (including the
1034 management of listener destinations and subscriptions) are WBEM server related requirements,
1035 and are typically implemented only once within an implementation.
- 1036 • Requirements that address the generation of indications are related to the referencing profile
1037 defining those indications, and are typically implemented as part of the implementation of that
1038 referencing profile.
- 1039 • Requirements that address functionality related to indication filters and filter collections are
1040 referencing profile related requirements.

1041 However, WBEM servers may contain other facilities allowing implementations of referencing
1042 profiles to delegate some of their implementation responsibilities to these facilities. For example,
1043 within WBEM servers providing a CIM instance repository the implementations of referencing
1044 profiles can delegate storing indication filters and filter collections to the CIM instance
1045 repository, such that in this case the implementation requirements for referencing profiles are
1046 effectively reduced to storing respective objects into the repository when the implementation of
1047 the referencing profile is installed.

1048 In this profile WBEM server related implementation requirements are marked with a phrase such as the
1049 following:

1050 "The requirements in this subclause are WBEM server related implementation requirements."

1051 In this profile referencing profile related implementation requirements are marked with a phrase such as
1052 the following:

1053 "The requirements in this subclause are referencing profile related implementation requirements."

1054 This facilitates explicit distinction of WBEM server related implementation requirements as opposed to
1055 requirements related to the implementation of referencing profiles.

1056 7.2 Features

1057 7.2.1 DynamicIndicationFilters

1058 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1059 The implementation of the DynamicIndicationFilters feature provides functionality for dynamic indication
1060 filters; for a description of dynamic indication filters, see 6.2.6.

1061 The granularity of the DynamicIndicationFilters feature is per IndicationService instance (see 7.3.2).

1062 The requirement level of the DynamicIndicationFilters feature is optional.

1063 The implementation of the DynamicIndicationFilters feature for a particular IndicationService instance is
1064 indicated by a value of True for the FilterCreationEnabled property.

1065 **7.2.2 IndicationServiceInitialSettingsExposed**

1066 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1067 The implementation of the IndicationServiceInitialSettingsExposed feature provides information about the
1068 initial settings of an indication service.

1069 The granularity of the IndicationServiceInitialSettingsExposed feature is per
1070 IndicationService instance (see 7.3.2).

1071 The requirement level of the IndicationServiceInitialSettingsExposed feature is optional.

1072 The availability of the IndicationServiceInitialSettingsExposed feature for a particular IndicationService
1073 instance is indicated by the presence of an IndicationServiceInitialSettings instance (see 7.3.9)
1074 associated through an InitialSettingsOfIndicationService instance (see 7.3.10).

1075 **7.2.3 IndicationServiceModification**

1076 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1077 The implementation of the IndicationServiceModification feature provides functionality for client requested
1078 dynamic modification of an indication service.

1079 The granularity of the IndicationServiceModification feature is per IndicationService instance (see 7.3.2).

1080 The requirement level of the IndicationServiceModification feature is optional.

1081 The availability of the IndicationServiceModification feature for a particular IndicationService instance is
1082 indicated if an IndicationServiceCapabilities (see 7.3.7) instance representing the capabilities of the
1083 represented indication service exists and is associated via the CapabilitiesOfIndicationService association
1084 (see 7.3.8), and in that instance the value True is set for any of the following properties:
1085 FilterCreationEnabledIsSettable, DeliveryRetryAttemptsIsSettable, DeliveryRetryIntervallsSettable,
1086 SubscriptionRemovalActionIsSettable, or SubscriptionRemovalTimeIntervallsSettable.

1087 **7.2.4 ReliableIndications**

1088 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1089 The implementation of the ReliableIndications feature provides functionality for reliable indications as
1090 described in 6.1.5. For further details, see 7.3.30 and 7.4.

1091 The granularity of the ReliableIndications feature is per IndicationService instance (see 7.3.2).

1092 The requirement level of the ReliableIndications feature is optional. The implementation of the
1093 ReliableIndications feature is also optional for listeners; in this case, the granularity is once per listener,
1094 and the discovery mechanism does not apply.

1095 The availability of the ReliableIndications feature for a particular IndicationService instance is indicated by
1096 a value larger than 0 for the DeliveryRetryAttempts property.

1097 **7.2.5 SuppressRepeatNotificationPolicy**

1098 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1099 The implementation of the SuppressRepeatNotificationPolicy feature provides functionality for
1100 suppressing repeated indication delivery by implementing the "suppress repeated indication delivery
1101 avoidance policy", as described in 6.1.6.3.

1102 The granularity of the SuppressRepeatNotificationPolicy feature is per implementation.

- 1103 The requirement level of the SuppressRepeatNotificationPolicy feature is optional.
- 1104 The availability of the SuppressRepeatNotificationPolicy feature is indicated by the value 3 (Suppress) for
1105 the RepeatNotificationPolicy property in AbstractSubscription instances (see 7.3.25) representing existing
1106 subscriptions.
- 1107 NOTE The discovery mechanism specified here is only rudimentary because the feature presence can only be
1108 discovered if at least one exploiting subscription is discovered. A future version of this profile is expected
1109 to introduce a new property into the CIM_IndicationServiceCapabilities class that indicates the presence of
1110 the feature per indication service.
- 1111 **7.2.6 DelayRepeatNotificationPolicy**
- 1112 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.
- 1113 The implementation of the DelayRepeatNotificationPolicy feature provides functionality for suppressing
1114 repeated indication delivery by implementing the "delayed indication delivery avoidance policy", as
1115 described in 6.1.6.4.
- 1116 The granularity of the DelayRepeatNotificationPolicy feature is per implementation.
- 1117 The requirement level of the DelayRepeatNotificationPolicy feature is optional.
- 1118 The availability of the DelayRepeatNotificationPolicy feature is indicated by the value 4 (Delay) for the
1119 RepeatNotificationPolicy property in AbstractSubscription instances (see 7.3.25) representing existing
1120 subscriptions.
- 1121 NOTE The discovery mechanism specified here is only rudimentary because the feature presence can only be
1122 discovered if at least one exploiting subscription is discovered. A future version of this profile is expected
1123 to introduce a new property into the CIM_IndicationServiceCapabilities class that indicates the presence of
1124 the feature per indication service.
- 1125 **7.2.7 IndividualFilterSubscription**
- 1126 The implementation of the IndividualFilterSubscription feature provides functionality for subscriptions to
1127 individual indication filters.
- 1128 The granularity of the IndividualFilterSubscription feature is per IndicationFilter instance (see 7.3.11).
- 1129 The requirement level of the IndividualFilterSubscription feature is optional.
- 1130 The availability of the IndividualFilterSubscription feature for a particular IndicationFilter instance is
1131 indicated by the value True for the IndividualSubscriptionSupported property.
- 1132 **7.2.8 FilterCollectionCoverageExposure**
- 1133 The implementation of the FilterCollectionCoverageExposure feature provides functionality for exposing
1134 the coverage of static filter collections.
- 1135 The granularity of the FilterCollectionCoverageExposure feature is per StaticFilterCollection instance (see
1136 7.3.17).
- 1137 The requirement level of the FilterCollectionCoverageExposure feature is optional.
- 1138 The availability of the FilterCollectionCoverageExposure feature for a particular StaticFilterCollection
1139 instance is indicated through at least one instance of either the IndicationFilterInFilterCollection
1140 association adaptation (see 7.3.19) or the FilterCollectionInFilterCollection association adaptation (see
1141 7.3.20) referencing the StaticFilterCollection instance.

1142 **7.2.9 LifeCycleGlobalIndicationFilter**

1143 The implementation of the LifeCycleGlobalIndicationFilter feature provides functionality for exposing a
1144 way to listen for a subset of life cycle indications.

1145 The granularity of the LifeCycleGlobalIndicationFilter feature is per implementation.

1146 The requirement level of the LifeCycleGlobalIndicationFilter feature is optional. Note that referencing
1147 profiles can require the LifeCycleGlobalIndicationFilter feature to be implemented.

1148 The availability of the LifeCycleGlobalIndicationFilter feature is indicated through the existence of the
1149 GlobalIndicationFilter (7.3.16) instances defined in 7.3.16.3.2.

1150 **7.2.10 AlertGlobalIndicationFilter**

1151 The implementation of the AlertGlobalIndicationFilter feature provides functionality for exposing a way to
1152 listen for a subset of life cycle indications.

1153 The granularity of the AlertGlobalIndicationFilter feature is per implementation.

1154 The requirement level of the AlertGlobalIndicationFilter feature is optional. Note that referencing profiles
1155 can require the AlertCycleGlobalIndicationFilter feature to be implemented.

1156 The availability of the AlertGlobalIndicationFilter feature is indicated through the existence of the
1157 GlobalIndicationFilter (7.3.16) instances defined in 7.3.16.3.1.

1158 **7.3 Adaptations**

1159 **7.3.1 Conventions**

1160 This profile repeats the effective values of certain Boolean qualifiers as part of property requirements, or
1161 of method parameter requirements. The following convention is established: If the name of a qualifier is
1162 listed, its effective value is True; if the qualifier name is not listed, its effective value is False. The
1163 convention is applied in the following cases:

- 1164 • In: indicates that the parameter is an input parameter
- 1165 • Out: indicates that the parameter is an output parameter
- 1166 • Key: indicates that the property is a key (that is, its value is part of the instance part)
- 1167 • Required: indicates that the element value shall be non-Null

1168 This profile defines operation requirements based on [DSP0223](#).

1169 For adaptations of ordinary classes and of associations the implementation requirements for operations
1170 are specified in adaptation-specific subclauses of 7.3.

1171 **7.3.2 IndicationService: CIM_IndicationService**

1172 **7.3.2.1 General**

1173 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1174 The IndicationService adaptation models indication services; indication services are described in 6.5.2.

1175 The implementation type of the IndicationService adaptation is: "instantiated".

1176 The IndicationService adaptation shall conform to the requirements for "central classes" defined in the
 1177 Profile Registration profile; for details, see [DSP1033](#).

1178 **7.3.2.2 Initial behavior**

1179 If the IndicationServiceInitialSettingsExposed feature (see 7.2.2) is implemented, the initial behavior of an
 1180 indication service shall be as exposed by the IndicationServiceInitialSettings instance (see 7.3.9) that is
 1181 associated with the IndicationService instance representing that indication service through an
 1182 InitialSettingsOfIndicationService instance (see 7.3.10).

1183 If the IndicationServiceInitialSettingsExposed feature (see 7.2.2) is not implemented, then the initial
 1184 behavior of the indication service shall be as follows:

- 1185 • Retry the delivery of an indication after a delivery failure three additional times, each time
 1186 waiting 20 seconds before the retry, and indicate this behavior with a value of 3 for the
 1187 DeliveryRetryAttempts property (see 7.3.2.3.3) and the value 20 for the DeliveryRetryInterval
 1188 property (see 7.3.2.3.4) in the IndicationService instance representing the indication service
- 1189 • Remove affected subscriptions after 30 days, and indicate this behavior with a value of 2
 1190 (Remove) for the SubscriptionRemovalAction property (see 7.3.2.3.5), and a value of 2,592,000
 1191 seconds (30 days) for the SubscriptionRemovalTimeInterval property (see 7.3.2.3.6) in the
 1192 IndicationService instance representing the indication service

1193 NOTE With respect to the availability of DynamicIndicationFilters feature (see 7.2.1) as indicated by the value of
 1194 the FilterCreationEnabled property an recommended initial behavior is not established; instead the
 1195 implementation is required to always expose the available behavior; see 7.3.2.3.2.

1196 **7.3.2.3 Element requirements**

1197 **7.3.2.3.1 General**

1198 Table 4 lists the element requirements for the IndicationService adaptation.

1199 **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.

Elements	Requirement	Description
SubscriptionRemovalTimeInterval	Mandatory	See 7.3.2.3.6.
Operations		
GetInstance()	Mandatory	See DSP0223 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	Mandatory	See DSP0223 .
ModifyInstance()	Conditional	See 7.3.2.3.7 and DSP0223 .

1200 If the ModifyInstance() operation is implemented (see 7.3.2.3.7), the values of some properties might be
 1201 modifiable through client requests; see 7.3.7 for details on indicating those properties whose values are
 1202 actually modifiable.

1203 **7.3.2.3.2 Property: FilterCreationEnabled**

1204 The value of the FilterCreationEnabled property shall reflect whether the DynamicIndicationFilters feature
 1205 (see 7.2.1) is available for the IndicationService instance. A value of False indicates that the feature is not
 1206 available; a value of True indicates that the feature is available.

1207 **7.3.2.3.3 Property: DeliveryRetryAttempts**

1208 The value of the DeliveryRetryAttempts property shall reflect the number of times that the implementation
 1209 is going to retry the delivery of an indication to a particular listener in the case of delivery failures. This
 1210 value does not include the initial delivery attempt.

1211 A value larger than 0 indicates that the ReliableIndications feature (see 7.2.4) is available. The value 0
 1212 indicates that the ReliableIndications feature is not available.

1213 **7.3.2.3.4 Property: DeliveryRetryInterval**

1214 The value of the DeliveryRetryInterval property shall reflect the minimal time interval in seconds that the
 1215 implementation waits before delivering an indication to a particular listener destination after a previous
 1216 delivery failure.

1217 **7.3.2.3.5 Property: SubscriptionRemovalAction**

1218 The value of the SubscriptionRemovalAction property shall reflect the removal action for subscriptions
 1219 after two failed indication deliveries where the time interval between the failed deliveries, without any
 1220 intermediate successful indication delivery, exceeds the timeout reflected by the value of the
 1221 SubscriptionRemovalTimeInterval property.

1222 **7.3.2.3.6 Property: SubscriptionRemovalTimeInterval**

1223 The value of the SubscriptionRemovalTimeInterval property shall reflect the minimum time interval that
 1224 implementations shall wait after two failed indication deliveries without any intermediate successful
 1225 indication delivery, before performing the activity designated by the value of the
 1226 SubscriptionRemovalAction property.

1227 **7.3.2.3.7 Method: ModifyInstance()**

1228 The implementation of the ModifyInstance() operation enables clients to modify aspects of the behavior
1229 of the represented indication service.

1230 The requirement level of the ModifyInstance() operation is conditional.

1231 Condition: The IndicationServiceModification feature is implemented; for a description, see 7.2.3.

1232 Information about which properties are modifiable is provided by an IndicationServiceCapabilities
1233 instance that is associated to the IndicationService instance representing the indication service; see 7.3.7
1234 and 7.3.8.

1235 Table 5 lists the error reporting requirements for the ModifyInstance() operation on IndicationService
1236 instances. If any of the error situations described in the Description column of Table 5 matches, the
1237 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
1238 reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1239 **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.
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.

1240 If the ModifyInstance() operation is successful, the requested modification on the indication service shall
1241 be applied, and — as a consequence — shall be reflected in all IndicationService instances that
1242 represent the modified indication service and are exposed by the implementation.

1243 If the ModifyInstance() operation fails, the requested modification on the indication service shall not be
1244 applied, and — as a consequence — all IndicationService instances that represent the indication service
1245 shall remain unchanged.

1246 **7.3.2.4 Instance requirements**

1247 Within an implementation there shall be exactly one indication service. That indication service shall be
 1248 represented by an IndicationService instance in the Interop namespace.

1249 NOTE 1 The reasons for requiring exactly one indication service are a) other elements defined in this profile (such
 1250 as subscriptions, listener destinations, or dynamic indication filters) require a relationship to the indication
 1251 service, and b) the modeled use of the CreateInstance() operation does not provide for expressing that
 1252 required relationship at creation time. For these reasons an indication service must be implied at creation
 1253 time, and the simplest approach for that is allowing just one indication service. Future versions of this
 1254 profile might lift the single instance restriction, for example by modeling respective creation methods with
 1255 parameters that enable establishing the required relationship to a specifiable indication service.

1256 NOTE 2 In some places in this profile multiple indication services are mentioned. This is not meant to lift the
 1257 restriction established in this subclause, but to accommodate the future introduction of multiple indication
 1258 services.

1259 **7.3.3 IndicationSystem: CIM_System**

1260 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1261 The IndicationSystem adaptation models indication systems; indication systems are described in 6.6.

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

1263 The IndicationSystem adaptation shall conform to the requirements for "scoping classes" defined in the
 1264 Profile Registration profile; for details, see [DSP1033](#).

1265 Table 6 lists the element requirements of the IndicationSystem adaptation.

1266 **Table 6 – IndicationSystem: Element requirements**

Elements	Requirement	Description
Properties		
Name	Mandatory	Key: See CIM schema definition.
CreationClassName	Mandatory	Key: See CIM schema definition.
Operations		
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	Mandatory	See DSP0223 .

1267 **7.3.4 HostedIndicationService: CIM_HostedService**

1268 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1269 The HostedIndicationService adaptation models the relationship between an indication service and its
1270 hosting indication system.

1271 The implementation type of the HostedIndicationService association adaptation is: "instantiated".

1272 Table 7 lists the element requirements for the HostedIndicationService association adaptation.

1273 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1274 Each IndicationSystem instance (see 7.3.3) shall be associated through a HostedIndicationService
1275 instance with the IndicationService instance (see 7.3.2) representing the indication service hosted by the
1276 indication system represented by the IndicationSystem instance.

1277 **7.3.5 IndicationsProfileRegistration: CIM_RegisteredProfile**

1278 **7.3.5.1 General**

1279 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1280 The IndicationsProfileRegistration adaptation models the profile registration of this profile, that is, the
1281 representation of the specific implemented version 1.2.2 of this profile.

1282 The implementation type of the IndicationsProfileRegistration adaptation is: "instantiated".

1283 The specific implemented version of this profile shall be represented by IndicationsProfileRegistration
1284 instances in the Interop namespace.

1285 NOTE The existence of an instance of this adaptation indicates that version 1.2.2 of this profile is implemented at
1286 least once within the WBEM server.

1287 Table 8 lists the element requirements for the IndicationsProfileRegistration adaptation.

1288

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 "1.2.2".
RegisteredOrganization	Mandatory	Value shall be 2 (DMTF).

1289 NOTE Operation requirements are defined by the base "profile class" CIM_RegisteredProfile defined in
 1290 [DSP1033](#).

1291 **7.3.6 ElementConformsToProfile: CIM_ElementConformsToProfile**

1292 The ElementConformsToProfile adaptation models the relationship between an indication service and the
 1293 profile registration of this profile (see 7.3.5).

1294 The implementation type of the ElementConformsToProfile association adaptation is: "instantiated".

1295 Table 9 lists the element requirements for the ElementConformsToProfile association adaptation.

1296

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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1297 Each IndicationService instance (see 7.3.2) shall be associated through an ElementConformsToProfile
 1298 instance with an IndicationsProfileRegistration instance (see 7.3.5).

1299 NOTE By requiring the implementation of the ElementConformsToProfile adaptation, this profile in fact requires
 1300 the central class profile advertisement methodology defined in [DSP1033](#). The scoping class profile
 1301 advertisement methodology is not applicable because the central instances of implementations of
 1302 referencing profiles will in almost all cases not be identical with the central instance of this profile, that is,
 1303 the IndicationSystem instance required by 7.3.3. Note that this does not restrict referencing profiles from
 1304 choosing a different methodology for their profile advertisement.

1305 **7.3.7 IndicationServiceCapabilities: CIM_IndicationServiceCapabilities**

1306 **7.3.7.1 General**

1307 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1308 The IndicationServiceCapabilities adaptation models the capabilities of indication services; indication
 1309 services are described in 6.5.2.

1310 The requirement level of the IndicationServiceCapabilities adaptation is conditional.

1311 Condition: The IndicationServiceModification feature is implemented; see 7.2.3.

1312 The implementation type of the IndicationServiceCapabilities adaptation is: "instantiated".

1313 **7.3.7.2 Element requirements**

1314 **7.3.7.2.1 General**

1315 Table 10 lists the element requirements for the IndicationServiceCapabilities adaptation.

1316 **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
SubscriptionRemovalTimeIntervalSettable	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

Element	Requirement	Description
Operations		
GetInstance()	Mandatory	See DSP0223 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	Mandatory	See DSP0223 .

1317 **7.3.7.2.2 Property: FilterCreationEnabledIsSettable**

1318 **DEPRECATED**

1319 The value of the FilterCreationEnabledIsSettable property shall indicate whether the implementation
 1320 supports modification of the FilterCreationEnabled property of the associated IndicationService instance.

1321 NOTE Values other than False are deprecated because it does not make sense enabling clients to set values of
 1322 properties that represent functionality that is either implemented or not implemented.

1323 **DEPRECATED**

1324 The value of the FilterCreationEnabledIsSettable property should be False, indicating that the
 1325 implementation does not support the modification of the FilterCreationEnabled property of the associated
 1326 IndicationService instance.

1327 **7.3.8 CapabilitiesOfIndicationService: CIM_ElementCapabilities**

1328 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1329 The CapabilitiesOfIndicationService adaptation models the relationship between an indication service and
 1330 its capabilities.

1331 The requirement level of the CapabilitiesOfIndicationService adaptation is conditional.

1332 Condition: The IndicationServiceModification feature is implemented; see 7.2.3.

1333 The implementation type of the CapabilitiesOfIndicationService association adaptation is: "instantiated".

1334 Table 11 lists the element requirements for the CapabilitiesOfIndicationService association adaptation.

1335 **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		

Elements	Requirement	Description
GetInstance()	Mandatory	See DSP0223 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1336 Each IndicationService instance (see 7.3.2) shall be associated through a CapabilitiesOfIndicationService
 1337 instance with at most one IndicationServiceCapabilities instance (see 7.3.7) representing the capabilities
 1338 of the indication service represented by the IndicationService instance.

1339 **7.3.9 IndicationServiceInitialSettings: CIM_IndicationServiceSettingData**

1340 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1341 The IndicationServiceInitialSettings adaptation models initial settings for indication services; indication
 1342 services are described in 6.5.2. The initial settings of an indication service are the settings that apply at
 1343 the point in time when the WBEM server hosting the indication service initially starts up the indication
 1344 service.

1345 The requirement level of the IndicationServiceInitialSettings adaptation is conditional.

1346 Condition: The IndicationServiceInitialSettingsExposed feature is implemented; see 7.2.2.

1347 The implementation type of the IndicationServiceInitialSettings adaptation is: "instantiated".

1348 Table 12 lists the element requirements for the IndicationServiceInitialSettings adaptation.

1349 **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.

Elements	Requirement	Description
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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	Mandatory	See DSP0223 .

1350 The initial settings of an indication service shall be represented by an IndicationServiceInitialSettings
 1351 instance in the Interop namespace.

1352 **7.3.10 InitialSettingsOfIndicationService: CIM_ElementSettingData**

1353 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1354 The InitialSettingsOfIndicationService association adaptation models the relationship between an
 1355 indication service and its initial settings; indication services are described in 6.5.2.

1356 The requirement level of the InitialSettingsOfIndicationService association adaptation is conditional.

1357 Condition: The IndicationServiceInitialSettingsExposed feature is implemented; see 7.2.2.

1358 The implementation type of the InitialSettingsOfIndicationService association adaptation is: "instantiated".

1359 Table 13 lists the element requirements for the InitialSettingsOfIndicationService association adaptation.

1360 **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 .

Elements	Requirement	Description
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1361 Each IndicationService instance (see 7.3.2) shall be associated through a
 1362 InitialSettingsOfIndicationService instance with at most one IndicationServiceInitialSettings instance (see
 1363 7.3.9) representing the initial settings of the indication service represented by the IndicationService
 1364 instance.

1365 **7.3.11 IndicationFilter: CIM_IndicationFilter**

1366 **7.3.11.1 General**

1367 The requirements in this subclause are referencing profile and WBEM server related implementation
 1368 requirements.

1369 The IndicationFilter adaptation models indication filters; indication filters are described in 6.2.

1370 The implementation type of the IndicationFilter adaptation is: "abstract".

1371 **7.3.11.2 Semantical requirements**

1372 For a particular indication filter the implementation shall filter any indication generated by (indication-
 1373 specific parts of) the implementation that is within the coverage of the indication filter, that is, that meets
 1374 both of the following requirements:

- 1375 • it matches the query statement (see 7.3.11.3.5) given by the value of the Query property in the
 1376 IndicationFilter instance representing the indication filter
- 1377 • its indication origin (see 6.1.2.4) is one of the local namespaces identified by the value of the
 1378 SourceNamespaces[] array property in that instance, or, in case that value is NULL, is the local
 1379 namespace in which the IndicationFilter instance representing the indication filter resides

1380 For the particular indication filter the implementation shall ignore any generated indication that does not
 1381 meet these requirements.

1382 Indications that passed an indication filter need to be further processed; see the requirements on the
 1383 IndicationFilterName property defined in 7.3.29.4.2, and the semantical requirements on listener
 1384 destinations defined in 7.3.23.2, and on subscriptions defined in 7.3.25.2. If implemented, the
 1385 requirements for reliable indications as defined in 7.3.30 and 7.4 may apply.

1386 Note that the indication filter semantics apply regardless of which profile specified the indications and
 1387 indication filters; thus an indication specified in one referencing profile is required to be considered by
 1388 indication filters specified in that referencing profile, but also by those specified in any other referencing
 1389 profile or in this profile and by those not specified in any profile.

1390 The indication filter semantics defined in this subclause do not require that an implementation implements
 1391 any of the indications within the coverage of an indication filter. However, referencing profiles may define
 1392 additional semantics for indication filters they define, including the case that the existence of a particular
 1393 IndicationFilter instance indicates that one or all indications within the coverage of the represented
 1394 indication filter are implemented. Of course, this approach is only feasible if the coverage covers one or
 1395 just a few indications.

1396 **7.3.11.3 Element requirements**

1397 **7.3.11.3.1 General**

1398 Table 14 lists the element requirements for the IndicationFilter adaptation.

1399 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	Mandatory	See DSP0223 .

1401 **7.3.11.3.2 Property: Name**

1402 The value of the Name property shall be the name of the indication filter; it shall be formatted as defined
 1403 by the following ABNF rule:

1404 `OrgID ":" RegisteredName ":" UniqueID`

1405 `OrgID` shall identify the business entity owning the referencing profile. `OrgID` shall include a copyrighted,
 1406 trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID
 1407 assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness,
 1408 `OrgID` shall not contain a colon (:). For referencing profiles owned by DMTF, `OrgID` shall match
 1409 "DMTF".

1410 `RegisteredName` shall be the registered name of the referencing profile, as defined by the value of the
 1411 `RegisteredName` property in the `RegisteredProfile` instance representing the implemented version of that
 1412 profile.

1413 `UniqueID` shall uniquely identify the represented indication filter within the referencing profile.

1414 DEPRECATED

1415 For compatibility with version 1.0 of this profile, referencing profiles owned by business entities other than
1416 DMTF may in addition define values for the Name property that are formatted as defined by the following
1417 ABNF rule:

1418 OrgID ":" UniqueID

1419 Where:

1420 OrgID is defined above in this subclause.

1421 UniqueID shall uniquely identify the instance within the business entity owning the referencing
1422 profile.

1423 Version 1.1 of this profile has deprecated this additional format.

1424 DEPRECATED

1425 7.3.11.3.3 Property: SourceNamespaces

1426 A non-Null value of this property is required for IndicationFilter instances in the Interop namespace; for
1427 IndicationFilter instances in other namespaces it is optional.

1428 If not Null, the value of the SourceNamespaces[] array property shall contain the names of local
1429 namespaces that are considered as potential indication origin namespaces (see 6.1.2.4) during indication
1430 filtering; see 7.3.11.2. The value shall not be an empty array.

1431 It is not required that the local namespaces identified by elements of value of the SourceNamespaces[]
1432 array property exist. If a non-existing local namespace is identified, no indications can originate out of that
1433 non-existing namespace; consequently, that element does not have an effect on indication filtering.
1434 However, if the identified namespace is added to the implementation at a later point in time, per the
1435 requirements of 7.3.11.2 indications originating out of that namespace are to be considered for indication
1436 filtering from then on.

1437 The value elements of the SourceNamespaces[] array property shall be formatted using the format that
1438 the implementation uses for value of the Name property in instances of the CIM_Namespace class that
1439 represent namespaces.

1440 7.3.11.3.4 Property: IndividualSubscriptionSupported

1441 The value of the IndividualSubscriptionSupported property shall be True if the IndividualFilterSubscription
1442 feature (see 7.2.7) is available for the IndicationFilter instance; otherwise, the value shall be False.

1443 7.3.11.3.5 Property: Query

1444 The value of the Query property shall be a properly formed query statement that is conformant to the
1445 requirements of the query language identified by the value of the QueryLanguage property, and that
1446 states the coverage of the indication filter.

1447 7.3.11.3.6 Property: QueryLanguage

1448 The value of the QueryLanguage property shall identify the query language in which the query statement
1449 exposed by the value of the Query property is expressed.

1450 NOTE This profile presently does not define a straight forward mechanism enabling clients to discover the set of
1451 query languages supported by an implementation. A future version of this profile is expected to introduce
1452 such a mechanism. For now, a rudimentary workaround may be inspecting the CIM representation of
1453 existing indication filters, thereby discovery a lower boundary for the set of supported query languages.

1454 **7.3.11.4 Instance requirements**

1455 Indication filters (see 6.2) shall be represented by IndicationFilter instances in the Interop namespace.

1456 The representation in namespaces other than the Interop namespace should be avoided. However, if
 1457 additional IndicationFilter instances represent an indication filter also in implementation namespaces,
 1458 these instances shall have the same key property values as the one in the Interop namespace.

1459 **7.3.12 StaticIndicationFilter: CIM_IndicationFilter**

1460 **7.3.12.1 General**

1461 The requirements in this subclause are referencing profile and WBEM server related implementation
 1462 requirements.

1463 The StaticIndicationFilter adaptation models static indication filters; static indication filters are described in
 1464 6.2.3.

1465 The implementation type of the StaticIndicationFilter adaptation is: "abstract".

1466 **7.3.12.2 Element requirements**

1467 **7.3.12.2.1 General**

1468 Table 15 lists the element requirements for the StaticIndicationFilter adaptation.

1469 **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_SUPPORTED.
DeleteInstance()	Prohibited	The implementation shall return the CIM status code CIM_ERR_NOT_SUPPORTED.
ModifyInstance()	Prohibited	The implementation shall return the CIM status code CIM_ERR_NOT_SUPPORTED.

1470 **7.3.12.2.2 Property: QueryLanguage**

1471 In adaptations based on the StaticIndicationFilter adaptation in referencing profiles owned by DMTF, the
 1472 value shall be "DMTF:CQL", thereby requiring CQL as the query language.

1473 **7.3.13 DynamicIndicationFilter: CIM_IndicationFilter**

1474 **7.3.13.1 General**

1475 The requirements in this subclause are WBEM server related implementation requirements.

1476 The DynamicIndicationFilter adaptation models dynamic indication filters; dynamic indication filters are
 1477 described in 6.2.6.

- 1478 The requirement level of the DynamicIndicationFilter adaptation is conditional.
- 1479 Condition: The DynamicIndicationFilters feature is implemented; see 7.2.1.
- 1480 The implementation type of the DynamicIndicationFilter adaptation is: "instantiated".

1481 **7.3.13.2 Element requirements**

1482 **7.3.13.2.1 General**

1483 Table 16 lists the element requirements for the DynamicIndicationFilter adaptation.

1484 **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.

1485 **7.3.13.2.2 Operation: CreateInstance()**

1486 Table 17 lists the error reporting requirements for the CreateInstance() operation on
 1487 DynamicIndicationFilter instances. If any of the error situations described in the Description column of
 1488 Table 17 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1489 addition, the error reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

1490 **Table 17 – CreateInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
CIM_ERR_INVALID_PARAMETER	Mandatory	The implementation is unable to support the filter name 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 namespaces 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 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.

Reporting mechanism	Requirement level	Description
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.

1491 If the CreateInstance() operation is successful, the requested dynamic indication filter shall be created,
 1492 and — as a consequence — shall be represented by a DynamicIndicationFilter instance in the requested
 1493 namespace.

1494 Clients should abstain from requesting the creation of DynamicIndicationFilter instances in namespaces
 1495 other than the Interop namespace. However, if the requested namespace is not the Interop namespace,
 1496 the implementation shall expose an additional DynamicIndicationFilter instance representing the dynamic
 1497 indication filter in the Interop namespace. That instance shall have identical values for all properties
 1498 except for the SourceNamespaces[] array property for which the provisions of 7.3.11.3.3 apply.

1499 If the CreateInstance() operation is fails, no dynamic indication filter shall be created, and — as a
 1500 consequence — no representing DynamicIndicationFilter instances shall be exposed in any namespace.

1501 **DEPRECATED**

1502 If the returned CIM status code is CIM_ERR_FAILED because an indication filter with the same coverage
 1503 as that requested already exists, the object_path of the CIM_IndicationFilter instance representing the
 1504 existing indication filter in the Interop namespace shall be returned as the value of the ErrorSource
 1505 property in the CIM_Error instance accompanying the CIM status code.

1506 NOTE Only this specific ad-hoc use of CIM_Error is deprecated. It is intended that a future version of this profile
 1507 introduces extended error handling based on standard error messages.

1508 **DEPRECATED**

1509 With respect to input values for key properties the rules defined in [DSP1001](#) apply, namely that
 1510 implementation may ignore any input value for non-reference key properties, and that clients should
 1511 abstain from providing input values for key properties.

1512 **7.3.13.2.3 Operation: DeleteInstance()**

1513 Table 18 lists the error reporting requirements for the DeleteInstance() operation on
 1514 DynamicIndicationFilter instances, and related CIM status codes. If any of the error situations described
 1515 in the Description column of Table 18 matches, the operation shall fail and the corresponding CIM status
 1516 code shall be returned. In addition, the error reporting requirements defined in [DSP0223](#) for the
 1517 DeleteInstance() operation apply.

1518 **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).

1519 If the DeleteInstance() operation succeeds, the represented dynamic indication filter shall be deleted and
 1520 — as a consequence — no longer be represented by any DynamicIndicationFilter instances in any
 1521 namespace exposed by the implementation.

1522 NOTE The instance requirements of associations representing relationships of the deleted dynamic indication
 1523 filter imply that respective association instances in any namespace exposed by the implementation cease
 1524 to exist; in this case this applies to IndicationServiceOfIndicationFilter instances (see 7.3.14). However,
 1525 note that the DeleteInstance() operation for the dynamic indication filter is required to fail if subscriptions
 1526 exist.

1527 If the DeleteInstance() operation fails, the dynamic indication filter shall not be deleted, and — as a
 1528 consequence — any representing DynamicIndicationFilter instances shall continue to exist as before.

1529 **7.3.13.2.4 Operation: ModifyInstance()**

1530 The implementation of the ModifyInstance() operation enables clients to modify aspects of the behavior
 1531 of the represented indication filter.

1532 The requirement level of the ModifyInstance() operation is optional.

1533 Table 19 lists the error reporting requirements for the ModifyInstance() operation on
 1534 DynamicIndicationFilter instances. If any of the error situations described in the Description column of
 1535 Table 19 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1536 addition, the error reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1537 **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.
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.

1538 If the ModifyInstance() operation is successful, the requested modification on the dynamic indication filter
 1539 shall be applied, and — as a consequence — shall be reflected in all DynamicIndicationFilter instances
 1540 that represent the modified dynamic indication filter and are exposed by the implementation.

1541 If the ModifyInstance() operation is fails, the requested modification on the dynamic indication filter shall
 1542 not be applied, and — as a consequence — all DynamicIndicationFilter instances that represent the
 1543 dynamic indication filter shall remain unchanged.

1544 **7.3.13.3 Instance requirements**

1545 Dynamic indication filters shall be represented by DynamicIndicationFilter instances; the additional
 1546 requirements of 7.3.11.4 apply.

1547 **7.3.14 IndicationServiceOfIndicationFilter: CIM_ServiceAffectsElement**

1548 The requirements in this subclause are referencing profile and WBEM server related implementation
 1549 requirements.

1550 The IndicationServiceOfIndicationFilter adaptation models the relationship between indication services
 1551 and the indication filters they manage.

1552 The implementation type of the IndicationServiceOfIndicationFilter association adaptation is:
 1553 "instantiated".

1554 Table 20 lists the element requirements for the IndicationServiceOfIndicationFilter association adaptation.

1555 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1556 Each IndicationService instance (see 7.3.2) shall be associated through an
 1557 IndicationServiceOfIndicationFilter instance with each IndicationFilter instance (see 7.3.11) representing
 1558 an indication filter managed by the indication service represented by the IndicationService instance.

1559 **7.3.15 IndicationSpecificIndicationFilter: CIM_IndicationFilter**

1560 **7.3.15.1 General**

1561 The requirements in this subclause are referencing profile and WBEM server related implementation
 1562 requirements.

1563 The IndicationSpecificIndicationFilter adaptation models indication-specific indication filters for indications
 1564 defined in referencing profiles or in this profile; indication-specific indication filters are described in 6.2.4.

- 1565 The requirement level of the `IndicationSpecificIndicationFilter` adaptation is optional.
- 1566 The `IndicationSpecificIndicationFilter` adaptation should be implemented if indications defined in a
1567 referencing profile or in this profile are implemented.
- 1568 The implementation type of the `IndicationSpecificIndicationFilter` adaptation is: "instantiated".

1569 **7.3.15.2 Element requirements**

1570 **7.3.15.2.1 General**

1571 Table 21 lists the element requirements for the `IndicationSpecificIndicationFilter` adaptation.

1572 **Table 21 – IndicationSpecificIndicationFilter: Element requirements**

Element	Requirement	Description
Base adaptations		
<code>StaticIndicationFilter</code>	Mandatory	See 7.3.12.
Properties		
Name	Mandatory	See 7.3.15.2.2.
Query	Mandatory	See 7.3.15.2.3.

1573 **7.3.15.2.2 Property: Name**

1574 The value of the Name property shall be formatted as defined by the following ABNF rule:

```
1575   OrgID ":" RegisteredName ":" IndicationAdaptationName "Filter" [ "/"
1576   MessageIdentification ]
```

1577 `OrgID` and `RegisteredName` shall be specified as detailed in 7.3.11.3.2.

1578 `IndicationAdaptationName` shall be the name of the indication adaptation defined in the profile
1579 identified by the `RegisteredName` rule. If the indication adaptation defines more than one possible
1580 indication.

1581 The `MessageIdentification` suffix only applies for the representation of indication-specific indication
1582 filters covering alert indications modeled by an adaptation based on the `AlertIndication` adaptation (see
1583 7.3.31); in this case for each alert indication defined by an alert message reference in the profile, a
1584 specific `IndicationSpecificIndicationFilter` instance is defined, where `MessageIdentification` shall be
1585 set as defined in 7.3.31.2 for the CIM representation of the alert indication. Thus, for alert indications,
1586 there is a one-to-one relationship between defined referenced alert messages and possible
1587 corresponding `IndicationSpecificIndicationFilter` instances.

1588 For lifecycle indications the suffix is not necessary because adaptations based on the `LifecycleIndication`
1589 adaptation (see 7.3.32) only can address one event, as defined by a (constant) query statement. Thus,
1590 for lifecycle indications, there is a one-to-one relationship between defined lifecycle indications and
1591 possible corresponding `IndicationSpecificIndicationFilter` instances.

1592 **7.3.15.2.3 Property: Query**

1593 The value of the Query property shall be identical with the event definition query statement (see 7.3.29.2)
1594 of the indication adaptation defined in the referencing profile or in this profile that is covered by the
1595 represented indication-specific indication filter. In the case `IndicationSpecificIndicationFilter` instances
1596 covering alert indications modeled by an adaptation based on the `AlertIndication` adaptation, the value of

1597 the Query property shall apply the ABNF rule named `EventQuerySingle` (see 7.3.31.2); that way for
 1598 alert indication adaptation referencing more than one alert message, separate
 1599 `IndicationSpecificIndicationFilter` instances are defined for each referenced alert message.

1600 **7.3.15.3 Instance requirements**

1601 If a profile defines an indication adaptation based on the `AlertIndication` adaptation (see 7.3.31) or the
 1602 `Lifecycle` adaptation (see 7.3.32), a corresponding indication-specific indication filter may be represented
 1603 by an `IndicationSpecificIndicationFilter` instance, with respective values of the `Name` and `Query`
 1604 properties.

1605 **NOTE** As with any indication filter (see 6.2.1), the existence of an indication-specific indication filter and its
 1606 representation by an `IndicationSpecificIndicationFilter` instance does not imply that the covered indication
 1607 is actually implemented. Furthermore, in the case where multiple implementations of the referencing profile
 1608 exist in a WBEM server, multiple `IndicationSpecificIndicationFilter` instances with identical values for `Name`
 1609 and `Query` properties may result.

1610 This profile leaves the decision whether or not to represent indication-specific indication filters as
 1611 `IndicationSpecificIndicationFilter` instances to the implementation; however, referencing profiles can
 1612 define an adaptation based on `IndicationSpecificIndicationFilter` adaptation that state more strict instance
 1613 requirements.

1614 In any case, if an implementation decides to represent indication-specific indication filters, these are to be
 1615 represented as required by the `IndicationSpecificIndicationFilter` adaptation. In addition, the requirements
 1616 of related adaptations such as the `ProfileSpecificFilterCollection` adaptation (see 7.3.21) or the
 1617 `IndicationFilterInFilterCollection` associations adaptation (see 7.3.19) apply.

1618 **7.3.16 GlobalIndicationFilter: CIM_IndicationFilter**

1619 **7.3.16.1 General**

1620 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1621 The `GlobalIndicationFilter` adaptation models global indication filters; global indication filters are described
 1622 in 6.2.5.

1623 The requirement level of the `GlobalIndicationFilter` adaptation is conditional.

1624 Condition: The `LifeCycleGlobalIndicationFilter` feature (see 7.2.9) or the `AlertGlobalIndicationFilter` feature
 1625 (see 7.2.10) is implemented.

1626 The implementation type of the `GlobalIndicationFilter` adaptation is: "instantiated".

1627 **7.3.16.2 Element requirements**

1628 Table 22 lists the element requirements for the `GlobalIndicationFilter` adaptation.

1629 **Table 22 – GlobalIndicationFilter: Element requirements**

Element	Requirement	Description
Base adaptations		
<code>StaticIndicationFilter</code>	Mandatory	See 7.3.12.

1630 **7.3.16.3 Instance requirements**

1631 **7.3.16.3.1 Instance requirements related to alert indications**

1632 Table 23 lists the property value requirements for GlobalIndicationFilter instances covering all alert
1633 indications.

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

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

1636 The requirement level of the instance requirements related to alert indications is conditional.

1637 Condition: The AlertGlobalIndicationFilter feature is implemented; see 7.2.10.

1638 **7.3.16.3.2 Instance requirements related to lifecycle indications**

1639 Table 24 lists the property value requirements for GlobalIndicationFilter instances covering all lifecycle
1640 indications of a particular subtype.

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

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"

1643 The requirement level of the instance requirements related to lifecycle indications is conditional.

1644 Condition: The LifeCycleGlobalIndicationFilter feature is implemented; see 7.2.9.

1645 **7.3.17 StaticFilterCollection: CIM_FilterCollection**

1646 **7.3.17.1 General**

1647 The requirements in this subclause are referencing profile and WBEM server related implementation
1648 requirements.

1649 The StaticFilterCollection adaptation models static filter collections; static filter collections are described in
1650 6.3.

1651 The implementation type of the StaticFilterCollection adaptation is: "abstract".

1652 **7.3.17.2 Semantical requirements**

1653 The coverage of a filter collection shall be the aggregated coverage of all the indication gates contained
1654 by the filter collection. This definition applies recursively to contained filter collections.

1655 NOTE Since filter collections aggregate the coverages of contained indication filters and contained other filter
1656 collections, and do not specify a filter query statement on their own, the defined coverage of a static filter
1657 collection is finally described by the set of query statements of its (directly or indirectly) aggregated
1658 indication filters.

- 1659 The implementation shall filter all indications generated by (indication-specific parts of) the
1660 implementation that are within the coverage of a filter collection.
- 1661 The implementation shall ignore any generated indication that is outside the coverage of the filter
1662 collection.
- 1663 If a particular indication is within the coverage of more than one indication gate contained by a filter
1664 collection, that indication shall pass the filter collection only once, and shall not be replicated for every
1665 matching contained indication gate.
- 1666 Indications that passed a filter collection need to be further processed; see the requirements on the
1667 IndicationFilterName property defined in 7.3.29.4.2, and the semantical requirements on listener
1668 destinations defined in 7.3.23.2, and on subscriptions defined in 7.3.25.2. If implemented, the
1669 requirements for reliable indications as defined in 7.3.30 and 7.4 may apply.
- 1670 These semantics apply regardless of whether all, some or no contained indication gates are represented
1671 as collection members in CIM. Thus clients and listeners need to be aware of the fact that the coverage of
1672 a static filter collection may be larger than that observable through inspection of CIM represented
1673 members of that static filter collection. In other words, indications could be delivered to subscribed
1674 listeners that are within the coverage of members of the static filter collection that are not currently
1675 represented in CIM; in the extreme case no members at all are CIM represented. On the other hand,
1676 even if the coverage of a static filter collection is not represented through CIM, clients may have a priori
1677 knowledge about the defined coverage of that static filter collection, for example by means of built-in
1678 program code or data; see 7.3.17.3.
- 1679 NOTE During runtime, the set of members of a static filter collection and the extent to which such members are
1680 represented in CIM may change. For example, consider the global filter collection with a defined coverage
1681 covering all alert indications defined in referencing profiles, as defined in 7.3.22.4.1. Its member set might
1682 grow or shrink over time as implementations of referencing profiles are installed in or removed from the
1683 implementation; however, the conceptual defined coverage of "all alert indications defined in referencing
1684 profile" remains constant.
- 1685 **7.3.17.3 Requirements pertaining to the defined coverage**
- 1686 For concrete adaptations based (directly or indirectly) on the StaticFilterCollection adaptation, profiles
1687 shall specify a defined coverage (see 6.3.3.3) through normative text that identifies indication filters
1688 and/or other filter collections as the *contained members* of the static filter collection, and thereby —
1689 because of 7.3.17.2 — as contributors to the coverage of the static filter collection.
- 1690 NOTE If in a chain of (abstract and concrete) adaptations based on the StaticFilterCollection adaptation the
1691 defined coverage is defined as part of an intermediate (abstract or concrete) adaptation, that definition
1692 propagates into adaptations (directly or indirectly) based on that intermediate adaptation.
- 1693 The defined coverage of a static filter collection always applies regardless of whether any members are
1694 represented in CIM. For contained static filter collections the specification of a defined coverage is
1695 likewise required.
- 1696 The definition of the defined coverage may be specified at the level of adaptations, or may be broken
1697 down to individual adaptation instances, or both.
- 1698 For examples of how to specify a defined coverage, see 7.3.21.3 and 7.3.22.
- 1699 **7.3.17.4 Element requirements**
- 1700 **7.3.17.4.1 General**
- 1701 Table 25 lists the element requirements for the StaticFilterCollection adaptation.

1702

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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	Mandatory	See DSP0223 .

1703 **7.3.17.4.2 Property: CollectionName**

1704 The value of the CollectionName property shall be formatted as defined by the following ABNF rule:

1705 `OrgID ":" RegisteredName ":" UniqueID`

1706 OrgID shall identify the business entity owning the referencing profile. OrgID shall include a copyrighted,
 1707 trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID
 1708 assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness,
 1709 OrgID shall not contain a colon (:).

1710 For referencing profiles owned by DMTF, OrgID shall match "DMTF".

1711 RegisteredName shall be the registered name of the referencing profile, as defined by the value of the
 1712 RegisteredName property in the RegisteredProfile instance representing the implemented version of the
 1713 referencing profile.

1714 UniqueID shall uniquely identify the instance within the implementation of the referencing profile.

1715 **DEPRECATED**

1716 For compatibility with version 1.0 of this profile, referencing profiles owned by business entities other than
 1717 DMTF may in addition define values for the CollectionName property that are formatted as defined by the
 1718 following ABNF rule:

1719 `OrgID ":" UniqueID`

1720 Where:

1721 OrgID is defined above in this subclause.

1722 UniqueID shall uniquely identify the instance within the business entity owning the referencing
 1723 profile.

1724 Version 1.1 of this profile has deprecated this additional format.

1725 **DEPRECATED**

1726 **7.3.17.5 Instance requirements**

1727 Static filter collections (see 6.3.3) shall be represented by StaticFilterCollection instances in the Interop
1728 namespace.

1729 The representation in namespaces other than the Interop namespace should be avoided. However, if
1730 additional StaticFilterCollection instances represent a static filter collection in implementation
1731 namespaces, these StaticFilterCollection instances shall have the same key property values as the one in
1732 the Interop namespace.

1733 If the FilterCollectionCoverageExposure feature (see 7.2.8) is available for a particular
1734 StaticFilterCollection instance, the contained members of the represented static filter collection (see
1735 7.3.17.3), and their containment relationship to the static filter collection are required to be represented in
1736 CIM; see 7.3.12 for the representation of contained static indication filters, see 7.3.17 for the
1737 representation of contained static filter collections, and see 7.3.19 and 7.3.20 for the representation of the
1738 containment relationship.

1739 **7.3.18 IndicationServiceOfFilterCollection: CIM_OwningCollectionElement**

1740 The requirements in this subclause are referencing profile and WBEM server related implementation
1741 requirements.

1742 The IndicationServiceOfFilterCollection adaptation models the relationship between a filter collection and
1743 the indication service that owns the filter collection.

1744 The implementation type of the IndicationServiceOfFilterCollection association adaptation is:
1745 "instantiated".

1746 Table 26 lists the element requirements for the IndicationServiceOfFilterCollection adaptation.

1747 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1748 Each IndicationService instance (see 7.3.2.4) shall be associated through an
1749 IndicationServiceOfFilterCollection instance to every StaticFilterCollection instance (see 7.3.17)
1750 representing a static filter collection managed by the indication service represented by the
1751 IndicationService instance.

1752 **7.3.19 IndicationFilterInFilterCollection: CIM_MemberOfCollection**

1753 The IndicationFilterInFilterCollection adaptation models the relationship between a filter collection and its
 1754 contained indication filters.

1755 The requirement level of the IndicationFilterInFilterCollection adaptation is conditional.

1756 Condition: The FilterCollectionCoverageExposure feature (see 7.2.8) is implemented.

1757 The implementation type of the IndicationFilterInFilterCollection association adaptation is: "instantiated".

1758 Table 27 lists the element requirements for the IndicationFilterInFilterCollection adaptation.

1759 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1760 Each StaticFilterCollection (see 7.3.17) instance shall be associated through an
 1761 IndicationFilterInFilterCollection instance with each of the IndicationFilter (see 7.3.11) instances
 1762 representing contained indication filters.

1763 **7.3.20 FilterCollectionInFilterCollection: CIM_MemberOfCollection**

1764 The requirements in this subclause are referencing profile and WBEM server related implementation
 1765 requirements.

1766 The FilterCollectionInFilterCollection adaptation models the relationship between a filter collection and its
 1767 contained other filter collections.

1768 The requirement level of the FilterCollectionInFilterCollection adaptation is conditional.

1769 Condition: All of the following:

- 1770 • The static filter collections in the managed environment are capable of containing other static
 1771 filter collections
- 1772 • The FilterCollectionCoverageExposure feature (see 7.2.8) is implemented.

1773 The implementation type of the FilterCollectionInFilterCollection association adaptation is: "instantiated".

1774 Table 28 lists the element requirements for the FilterCollectionInFilterCollection adaptation.

1775

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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

1776 Each StaticFilterCollection instance (see 7.3.17) representing a static filter collection that contains other
 1777 static filter collections shall be associated through a FilterCollectionInFilterCollection instance with each of
 1778 the StaticFilterCollection instances (see 7.3.17) representing a contained static filter collection.

1779 **7.3.21 ProfileSpecificFilterCollection: CIM_FilterCollection**

1780 **7.3.21.1 General**

1781 The requirements in this subclause are referencing profile and WBEM server related implementation
 1782 requirements.

1783 The ProfileSpecificFilterCollection adaptation models profile-specific filter collections; profile-specific filter
 1784 collections are described in 6.3.3.4.

1785 The requirement level of the ProfileSpecificFilterCollection adaptation is optional.

1786 The ProfileSpecificFilterCollection adaptation should be implemented.

1787 The implementation type of the ProfileSpecificFilterCollection adaptation is: "instantiated".

1788 **7.3.21.2 Element requirements**

1789 **7.3.21.2.1 General**

1790 Table 29 lists the element requirements for the ProfileSpecificFilterCollection adaptation.

1791

Table 29 – ProfileSpecificFilterCollection: Element requirements

Element	Requirement	Description
Base adaptations		
StaticFilterCollection	Mandatory	See 7.3.17.
Properties		
CollectionName	Mandatory	See 7.3.21.2.2.

1792 7.3.21.2.2 Property: CollectionName

1793 The value of the CollectionName property shall be formatted as defined by the following ABNF rule:

```
1794   OrgID ":" RegisteredName ":"  
1795   "ProfileSpecified" Type "IndicationFilterCollection"
```

1796 OrgID and RegisteredName shall be specified as detailed in 7.3.17.4.2.

1797 Type shall be "Alert" in case the represented profile-specific filter collection covers all alert indications,
1798 and shall be "Lifecycle" in case the represented profile-specific filter collection covers all lifecycle
1799 indications defined in the referencing profile identified by RegisteredName.

1800 NOTE This requirement does not preclude more than one instance in the Interop namespace from having
1801 identical values for the CollectionName property, because, for example, the referencing profile could be
1802 implemented more than once.

1803 7.3.21.3 Requirements pertaining to the defined coverage

1804 Requirements pertaining to the defined coverage are specified on a per instance basis; see 7.3.21.4
1805 and 7.3.21.4.2.

1806 7.3.21.4 Instance requirements**1807 7.3.21.4.1 Instance requirements for profile-specific filter collections covering all alert indications
1808 specified in a profile**

1809 If and only if a referencing profile defines alert indications, the implementation may expose a
1810 ProfileSpecificFilterCollection instance in the Interop namespace that covers all alert indications defined
1811 in that profile. The element requirements defined in 7.3.21.2 apply.

1812 NOTE The existence of that ProfileSpecificFilterCollection instance does not imply that any alert indications are
1813 actually implemented. Furthermore, in the case where multiple implementations of the referencing profile
1814 exist in a WBEM server, multiple ProfileSpecificFilterCollection instances may result.

1815 The members of a profile-specific filter collection covering all alert indications defined in a referencing
1816 profile shall be all indication-specific indication filters covering the alert indications defined in that
1817 referencing profile; see 7.3.15. This definition in effect defines the defined coverage as all alert indications
1818 defined in the referencing profile.

1819 NOTE For existing ProfileSpecificFilterCollection instances the instance requirements of association instances
1820 representing relationships of the represented profile-specific filter collection apply; for example, see 7.3.18,
1821 7.3.19 or 7.3.20.

**1822 7.3.21.4.2 Instance requirements for profile-specific filter collections covering all lifecycle
1823 indications specified in a profile**

1824 If and only if a referencing profile defines lifecycle indications, the implementation may expose a
1825 ProfileSpecificFilterCollection instance in the Interop namespace that covers all lifecycle indications
1826 defined in that profile. The element requirements defined in 7.3.21.2 apply.

1827 NOTE The existence of such a ProfileSpecificFilterCollection instance does not imply that any lifecycle indications
1828 are actually implemented. Furthermore, in the case where multiple implementations of the referencing
1829 profile exist in a WBEM server, multiple ProfileSpecificFilterCollection instances may result.

1830 The members of a profile-specific filter collection covering all lifecycle indications defined in a referencing
1831 profile shall be all indication-specific indication filters covering the lifecycle indications defined in that
1832 referencing profile or in this profile; see 7.3.15. This definition in effect defines the defined coverage as all
1833 lifecycle indications defined in the referencing profile.

1834 NOTE For existing ProfileSpecificFilterCollection instances the instance requirements of association instances
 1835 representing relationships of the represented profile-specific filter collection apply; for example, see 7.3.18,
 1836 7.3.19 or 7.3.20.

1837 The requirements specified in this subclause for lifecycle indications defined in referencing profiles shall
 1838 also apply for the lifecycle indications defined in this profile; see 7.3.33 and 7.3.34.

1839 **7.3.22 GlobalFilterCollection: CIM_FilterCollection**

1840 **7.3.22.1 General**

1841 The requirements in this subclause are referencing profile and WBEM server related implementation
 1842 requirements; see 7.1.

1843 The GlobalFilterCollection adaptation models global filter collection; global filter collections are described
 1844 in 6.3.3.5.

1845 The implementation type of the GlobalFilterCollection adaptation is: "instantiated".

1846 **7.3.22.2 Element requirements**

1847 Table 30 lists the element requirements for the GlobalFilterCollection adaptation.

1848 **Table 30 – GlobalFilterCollection: Element requirements**

Element	Requirement	Description
Base adaptations		
StaticFilterCollection	Mandatory	See 7.3.17.

1849 **7.3.22.3 Requirements pertaining to the defined coverage**

1850 Requirements pertaining to the defined coverage are specified on a per instance basis; see 7.3.22.4.1,
 1851 7.3.22.4.2, 7.3.22.4.3 and 7.3.22.4.4.

1852 **7.3.22.4 Instance requirements**

1853 **7.3.22.4.1 Instance requirements for the global filter collection covering all alert indications**
 1854 **specified in profiles**

1855 If any alert indications specified in referencing profiles or in this profile are implemented, the
 1856 implementation may expose a GlobalFilterCollection instance in the Interop namespace that covers all
 1857 alert indications defined in profiles. In implementations where it is not possible to determine whether alert
 1858 indications specified in referencing profiles are implemented, the instance may be exposed if the delivery
 1859 of alert indications is implemented in general.

1860 In the GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
 1861 following ABNF rule:

1862 "DMTF:Indications:"
 1863 "GlobalProfileSpecifiedAlertIndicationFilterCollection".

1864 In this case the members of the represented global filter collection shall be all profile-specific filter
 1865 collections covering the alert indications defined in any implemented referencing profile or in this profile;
 1866 see 7.3.21.4. This definition in effect specifies the defined coverage as all alert indications defined in
 1867 referencing profiles and in this profile; if instantiated by an implementation, the coverage would be all
 1868 implemented alert indications out of that set.

1869 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
1870 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
1871 or 7.3.20.

1872 **7.3.22.4.2 Instance requirements for the global filter collection covering all lifecycle indications** 1873 **specified in profiles**

1874 If any lifecycle indications specified in referencing profiles or in this profile are implemented, the
1875 implementation may expose a GlobalFilterCollection instance in the Interop namespace that covers all
1876 lifecycle indications defined in profiles. In implementations where it is not possible to determine whether
1877 lifecycle indications specified in referencing profiles are implemented, the instance may be exposed if the
1878 delivery of lifecycle indications is implemented in general.

1879 In GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
1880 following ABNF rule:

```
1881 "DMTF:Indications:"  
1882 "GlobalProfileSpecifiedLifecycleIndicationFilterCollection".
```

1883 The members of the represented global filter collection shall be all profile-specific filter collections
1884 covering the lifecycle indications defined in any implemented referencing profile or in this profile; see
1885 7.3.21.4.2. This definition in effect specifies the defined coverage as all lifecycle indications defined in
1886 referencing profiles and in this profile; if instantiated by an implementation, the coverage would be all
1887 implemented lifecycle indications out 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 **7.3.22.4.3 Instance requirements for the global filter collection covering all indications specified** 1892 **in profiles**

1893 If any indications specified in referencing profiles or in this profile are implemented, the implementation
1894 may expose a GlobalFilterCollection instance in the Interop namespace that covers all indications defined
1895 in profiles. In implementations where it is not possible to determine whether indications specified in
1896 referencing profiles are implemented, the instance may be exposed if the delivery of indications is
1897 implemented in general.

1898 In the GlobalFilterCollection instance, the value of the CollectionName property shall be as defined by the
1899 following ABNF rule:

```
1900 "DMTF:Indications:"  
1901 "GlobalProfileSpecifiedIndicationFilterCollection"
```

1902 The members of the represented global filter collection shall be the following global filter collections (if
1903 existing):

- 1904 • the global filter collection covering all alert indications defined in any implemented referencing
1905 profile, as required in 7.3.22.4.1
- 1906 • the global filter collection covering all lifecycle indications defined in any implemented
1907 referencing profile, as required in 7.3.22.4.2

1908 This definition in effect specifies the defined coverage as all indications defined in referencing profiles and
1909 in this profile; if instantiated by an implementation, the coverage would be all implemented indications out
1910 of that set.

1911 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
1912 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
1913 or 7.3.20.

1914 **7.3.22.4.4 Instance requirements for the global filter collection covering all lifecycle indications**

1915 If the implementation supports the delivery of lifecycle indications, the implementation shall expose a
 1916 GlobalFilterCollection instance in the Interop namespace that covers all lifecycle indications defined in
 1917 profiles.

1918 In GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
 1919 following ABNF rule:

1920 `"DMTF:Indications:GlobalLifecycleIndicationFilterCollection".`

1921 The members of the represented global filter collection shall be all profile-specific filter collections
 1922 covering the global indication filters that each cover all indications of one of the three subtypes of lifecycle
 1923 indications (CIM_InstCreation, CIM_InstDeletion and CIM_InstModification); see 7.3.16.3.2.

1924 This definition in effect specifies the defined coverage as all lifecycle indications defined in referencing
 1925 profiles and in this profile.

1926 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
 1927 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
 1928 or 7.3.20.

1929 **7.3.23 ListenerDestination: CIM_ListenerDestination**

1930 **7.3.23.1 General**

1931 The ListenerDestination adaptation models listener destinations; listener destinations are described in
 1932 6.4.5.

1933 The implementation type of the ListenerDestination adaptation is: "instantiated".

1934 **7.3.23.2 Semantical requirements**

1935 For a particular listener destination, an implementation shall deliver any indication that passed the
 1936 indication gate (see 6.2 or 6.3) referenced by any subscription (see 6.4.1) that also references the listener
 1937 destination, to the listener referenced by that listener destination. See also the semantical requirements
 1938 on indication filters defined in 7.3.11.2, on filter collections defined in 7.3.17.2, and on subscriptions
 1939 defined in 7.3.25.2.

1940 NOTE It is possible that a particular indication is delivered more than once to a particular listener for various
 1941 reasons, such as that the listener is referenced by more than one listener destination, or that the indication
 1942 is within the coverage of more than one indication gate, each of which is referenced by a subscription
 1943 referencing the listener destination referencing the listener.

1944 **7.3.23.3 Element requirements**

1945 **7.3.23.3.1 General**

1946 Table 31 lists the element requirements of the ListenerDestination adaptation.

1947 **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.

Element	Requirement	Description
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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .
Associators()	Mandatory	See DSP0223 .
AssociatorNames()	Mandatory	See DSP0223 .
References()	Mandatory	See DSP0223 .
ReferenceNames()	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 .

1948 **7.3.23.3.2 Property: Destination**

1949 The value of the Destination property shall identify the listener referenced by the listener destination.

1950 A value of Null for the Destination property indicates a free listener destination (see 6.4.5).

1951 If the value of the Destination property is not Null, it shall be a valid IETF Uniform Resource Identifier
1952 value (as defined in [RFC3986](#)) including the scheme, host and port as part of the URI Location.

1953 **7.3.23.3.3 Property: PersistenceType**

1954 The value of the PersistenceType property shall describe the durability of the represented listener
1955 destination.

1956 The property values shall be constrained to 3 (Transient), 2 (Permanent), and Null.

1957 If the listener destination is permanent, then the value of the PersistenceType property shall be either Null
1958 or 2 (Permanent). Permanent listener destinations are long-lived and are expected to be available for
1959 indication delivery. For example, a typical listener referenced by a permanent listener destination would
1960 be a system log file. The inability of an implementation to deliver indications to a listener referenced by a
1961 permanent listener destination will be treated as an error condition by the implementation, as defined in
1962 7.4.3.5.

1963 If the listener destination is transient, then the value of the PersistenceType property shall be 3
1964 (Transient). Transient listener destinations are short-lived and have less strong requirements (than
1965 permanent listener destinations) regarding their availability for indication delivery. For example, a typical
1966 listener referenced by a transient listener destination would be a task progress meter in a graphical
1967 management application. The inability of an implementation to deliver indications to a listener described
1968 by a transient listener destination will be handled by removing the listener destination and its
1969 subscriptions from the implementation, as defined in 7.4.3.6.

1970 **7.3.23.3.4 Operation: CreateInstance()**

1971 Table 32 lists the error reporting requirements for the CreateInstance() operation on ListenerDestination
1972 instances. If any of the error situations described in the Description column of Table 32 matches, the

1973 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1974 reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

1975 **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.

1976 If the CreateInstance() operation is successful, the requested listener destination shall be created, and —
 1977 as a consequence — shall be represented by a ListenerDestination instance in the requested
 1978 namespace. In addition, if the requested namespace is not the Interop namespace, the implementation
 1979 shall expose an additional ListenerDestination instance representing the listener destination in the Interop
 1980 namespace (see 7.3.23.4).

1981 If the CreateInstance() operation fails, no listener destination shall be created, and — as a consequence
 1982 — no representing ListenerDestination instances shall be exposed in any namespace.

1983 The implementation may ignore the values of key properties in the embedded CIM_ListenerDestination
 1984 instance passed as the value of the NewInstance parameter.

1985 Clients should abstain from providing the values of key properties in the embedded
 1986 CIM_ListenerDestination instance passed as the value of the NewInstance parameter.

1987 Clients should abstain from requesting the creation of ListenerDestination instances in namespaces other
 1988 than the Interop namespace.

1989 Clients should favor the re-use of an existing listener destination referencing a particular listener over the
 1990 creation of a new listener destination referencing the same listener.

1991 **7.3.23.3.5 Operation: DeleteInstance()**

1992 Table 33 lists the error reporting requirements for the DeleteInstance() operation on ListenerDestination
 1993 instances, and related CIM status codes. If any of the error situations described in the Description column
 1994 of Table 33 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1995 addition, the error reporting requirements defined in [DSP0223](#) for the DeleteInstance() operation apply.

1996 **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).

- 1997 If the DeleteInstance() operation is successful, the represented listener destination shall be deleted and
- 1998 — as a consequence — shall no longer be represented by ListenerDestination instances in any
- 1999 namespace exposed by the implementation.
- 2000 NOTE The instance requirements of associations representing relationships of the deleted listener destination
- 2001 imply that respective association instances in any namespace exposed by the implementation cease to
- 2002 exist; in this case this applies to IndicationServiceOfListenerDestination instances (see 7.3.24). However,
- 2003 note that the DeleteInstance() operation for the listener destination is required to fail if subscriptions exist.
- 2004 If the DeleteInstance() operations fails, the listener destination shall not be deleted, and — as a
- 2005 consequence — any representing ListenerDestination instances shall continue to exist as before.
- 2006 **7.3.23.3.6 Operation: ModifyInstance()**
- 2007 The ModifyInstance operation may be available for an instance of CIM_ListenerDestination.
- 2008 The implementation of the ModifyInstance() operation enables clients to modify existing listener
- 2009 destinations.
- 2010 The requirement level of the ModifyInstance() operation is optional.
- 2011 Table 34 lists the error reporting requirements for the ModifyInstance() operation on ListenerDestination
- 2012 instances. If any of the error situations described in the Description column of Table 34 matches, the
- 2013 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
- 2014 reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

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).

- 2016 If the ModifyInstance() operation is successful, the requested modification on the listener destination
- 2017 shall be applied, and — as a consequence — shall be reflected in all ListenerDestination instances that
- 2018 represent the modified listener destination and are exposed by the implementation.
- 2019 If the ModifyInstance() operation fails, the requested modification on the listener destination shall not be
- 2020 applied, and — as a consequence — all ListenerDestination instances that represent the listener
- 2021 destination shall remain unchanged.

2022 **7.3.23.4 Instance requirements**

2023 Listener destinations (see 6.4.5) shall be represented by ListenerDestination instances in the Interop
2024 namespace.

2025 The representation in namespaces other than the Interop namespace should be avoided. However, if
2026 additional ListenerDestination instances represent the listener destination in implementation namespaces,
2027 these ListenerDestination instances shall have the same key property values as the one in the Interop
2028 namespace.

2029 **7.3.24 IndicationServiceOfListenerDestination: CIM_ServiceAffectsElement**

2030 The IndicationServiceOfListenerDestination adaptation models the relationship between indication
2031 services and the listener destinations they manage. Indication services are described in 6.5.2; listener
2032 destinations are described in 6.4.5.

2033 The implementation type of the IndicationServiceOfListenerDestination association adaptation is:
2034 "instantiated".

2035 Table 35 lists the elements requirements of the IndicationServiceOfListenerDestination adaptation.

2036 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

2037 Each IndicationService (see 7.3.2) instance shall be associated through an
2038 IndicationServiceOfListenerDestination instance with each ListenerDestination (see 7.3.23) instance
2039 representing a listener destination managed by the indication service represented by the
2040 IndicationService instance.

2041 **7.3.25 AbstractSubscription: CIM_AbstractIndicationSubscription**

2042 **7.3.25.1 General**

2043 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2044 The AbstractSubscription adaptation models subscriptions for the delivery of indications from an
2045 indication gate to a listener referenced by a listener destination; subscriptions are described in 6.4.

2046 The implementation type of the AbstractSubscription association adaptation is: "abstract".

2047 **7.3.25.2 Semantical requirements**

2048 An implementation shall deliver any indication that passed the indication gate referenced by the
 2049 subscription (that is, any indication generated by the implementation that is within the coverage of the
 2050 indication gate) to the listener referenced by the listener destination referenced by the subscription.

2051 A listener that is referenced by the listener destination referenced by a subscription needs to be prepared
 2052 to receive any indication that is within the coverage of the indication gate referenced by that subscription.
 2053 Of course, listeners may ignore received indications.

2054 **7.3.25.3 Element requirements**

2055 Table 36 lists the element requirements for the AbstractSubscription adaptation.

2056 **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.
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.

Elements	Requirement	Description
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.		

2057 **7.3.25.3.1 Property: OnFatalErrorPolicy**

2058 The value of the OnFatalErrorPolicy property shall indicate the behavior that the implementation exposes
 2059 with respect to represented subscriptions in case of failures that imply that some aspect of indication
 2060 generation processing or indication delivery is no longer functioning and indications may be lost.

2061 A value of 4 (Remove) shall indicate that the implementation performs implicit subscription removal as
 2062 detailed in 7.4.3.6; this shall be the default behavior.

2063 **7.3.25.3.2 Property: RepeatNotificationPolicy**

2064 The value of the RepeatNotificationPolicy property shall indicate the policy that the implementation
 2065 applies with respect to the avoidance of repeated indication delivery of repeated indications as described
 2066 in 6.1.6.

2067 Table 37 lists constraints for the value of the RepeatNotificationPolicy property.

2068 **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)

2069 **7.3.25.3.3 Property: RepeatNotificationInterval**

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

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

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 RepeatNotificationInterval property shall be the length of the time interval in seconds that the
 2076 implementation waits after initial delivery of a number of repeated indications as indicated by the value of
 2077 the RepeatNotificationCount property before delivering the next repeated indication.

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 RepeatNotificationInterval property shall be the length of the monitoring time interval in seconds during
 2081 which the implementation monitors the indication gate referenced by the subscription for a number of
 2082 additional repeated indications. Furthermore, only if during that monitoring interval at least the number of
 2083 repeated indications as indicated by the value of the RepeatNotificationCount accrue, delivers only the
 2084 first indication as a substitute for all the repeated indications accrued during the monitoring time interval.

2085 7.3.25.3.4 Property: RepeatNotificationGap

2086 The requirement level of the RepeatNotificationGap property is conditional exclusive.

2087 Condition: The DelayRepeatNotificationPolicy feature (see 7.2.6) is implemented.

2088 The value of the RepeatNotificationGap property shall be the length of the delay time interval in seconds
2089 that the implementation waits after delivering the first of a number of repeated indications that accrued
2090 during the monitoring time interval, before starting another monitoring time interval, as described in
2091 7.3.25.3.5 with respect to implementations of the DelayRepeatNotificationPolicy feature.

2092 7.3.25.3.5 Property: RepeatNotificationCount

2093 The requirement level of the RepeatNotificationCount property is conditional exclusive.

2094 Condition: Either the SuppressRepeatNotificationPolicy feature (see 7.2.5) or the
2095 DelayRepeatNotificationPolicy feature (see 7.2.6) is implemented.

2096 If the implementation applies the SuppressRepeatNotificationPolicy feature (see 7.2.5) for the
2097 represented subscription, as indicated by the value 3 (Suppress) for the RepeatNotification property, the
2098 value of the RepeatNotificationCount property shall be the number of repeated indications that the
2099 implementation delivers before suppressing the delivery of further repeated indications within the time
2100 interval exposed by the value of the RepeatNotificationInterval property.

2101 If the implementation applies the DelayRepeatNotificationPolicy feature (see 7.2.6) for the represented
2102 subscription, as indicated by the value 4 (Delay) for the RepeatNotification property, the value of the
2103 RepeatNotificationCount property shall be the number of repeated indications that the implementation is
2104 required to monitor and delay during the monitoring time interval exposed by the value of the
2105 RepeatNotificationInterval property. Only if during that monitoring time interval the number of accrued
2106 repeated indications reaches that number, the implementation shall deliver the first of repeated indication
2107 as a substitute for the accrued repeated indications. In other words, the quotient of the values of the
2108 RepeatNotificationCount and the RepeatNotificationInterval properties expresses a rate of repeated
2109 indications that must have been reached or exceeded during the monitoring time interval before one
2110 indication is delivered at the end of the monitoring time interval.

2111 7.3.25.3.6 Operation: DeleteInstance()

2112 The error situations and CIM status codes defined in [DSP0223](#) for the DeleteInstance() operation apply.

2113 If the DeleteInstance() operation succeeds, the represented subscription shall be deleted and — as a
2114 consequence — shall no longer be represented by any AbstractSubscription instances in any namespace
2115 exposed by the implementation.

2116 If the DeleteInstance() operation fails, the subscription shall not be deleted, and — as a consequence —
2117 any representing AbstractSubscription instances shall continue to exist as before.

2118 7.3.25.3.7 Operation: ModifyInstance()

2119 The requirement level of the ModifyInstance() operation is optional.

2120 Table 38 lists the error reporting requirements for the ModifyInstance() operation on AbstractSubscription
2121 instances, and related CIM status codes. If any of the error situations described in the Description column
2122 of Table 38 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
2123 addition, the error reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation are
2124 applicable.

2125

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.

2126 If the ModifyInstance() operation is successful, the requested modification on the represented
 2127 subscription shall be applied, and — as a consequence — shall be reflected in all AbstractSubscription
 2128 instances that represent the modified subscription.

2129 If the ModifyInstance() operation fails, the requested modification on the subscription shall not be
 2130 applied, and — as a consequence — all AbstractSubscription instances that represent the subscription
 2131 shall remain unchanged.

2132 **7.3.25.4 Instance requirements**

2133 Subscriptions (see 6.4.1) shall be represented by AbstractSubscription instances in the Interop
 2134 namespace that relate either IndicationFilter instances (see 7.3.11) or StaticFilterCollection instances
 2135 (see 7.3.17) with ListenerDestination instances (see 7.3.23).

2136 The representation in namespaces other than the Interop namespace should be avoided. However, if
 2137 both the indication filter/filter collection and the related listener destination represented by the referenced
 2138 instances in the Interop namespace are also represented by additional instances in other namespaces,
 2139 respective AbstractSubscription instances shall represent the subscription in these other namespaces as
 2140 well.

2141 **7.3.26 FilterSubscription: CIM_IndicationSubscription**

2142 **7.3.26.1 General**

2143 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2144 The FilterSubscription adaptation models subscriptions for the delivery of indications from an indication
 2145 filter to a listener referenced by a listener destination; subscriptions are described in 6.4.

2146 The requirement level of the FilterSubscription adaptation is conditional.

2147 Condition: The IndividualFilterSubscription feature (see 7.2.7) is implemented.

2148 The implementation type of the FilterSubscription association adaptation is: "instantiated".

2149 **7.3.26.2 Semantical requirements**

2150 The semantical requirements of 7.3.25.2 apply respectively for the FilterSubscription adaptation.

2151 **7.3.26.3 Element requirements**

2152 **7.3.26.3.1 General**

2153 Table 39 lists the element requirements for the FilterSubscription adaptation.

2154 **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 .

2155 **7.3.26.3.2 Operation: CreateInstance()**

2156 Table 40 lists the error reporting requirements for the CreateInstance() operation on FilterSubscription
 2157 instances. If any of the error situations described in the Description column of Table 40 matches, the
 2158 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 2159 reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

2160 **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.

Reporting mechanism	Requirement level	Description
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.		

2161 If the CreateInstance() operation is successful, the requested filter subscription was created, and
 2162 consequently — as required by 7.3.26.4 — shall be represented by a FilterSubscription instance in the
 2163 requested namespace. In addition, if the requested namespace is not the Interop namespace, the
 2164 implementation shall expose an additional FilterSubscription instance representing the subscription in the
 2165 Interop namespace (see 7.3.26.4).

2166 If the CreateInstance() operation fails, no subscription shall be created, and — as a consequence — no
 2167 representing FilterSubscription instances shall be exposed in any namespace.

2168 Clients should abstain from requesting the creation of FilterSubscription instances in namespaces other
 2169 than the Interop namespace.

2170 **7.3.26.4 Instance requirements**

2171 The requirements of 7.3.25.4 apply respectively for FilterSubscription instances.

2172 **7.3.27 CollectionSubscription: CIM_FilterCollectionSubscription**

2173 **7.3.27.1 General**

2174 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2175 The CollectionSubscription adaptation models subscriptions for the delivery of indications from a filter
 2176 collection to a listener referenced by a listener destination; subscriptions are described in 6.4.

2177 The implementation type of the FilterCollectionSubscription association adaptation is: "instantiated".

2178 **7.3.27.2 Semantical requirements**

2179 The semantical requirements of 7.3.25.2 apply respectively for the CollectionSubscription adaptation.

2180 **7.3.27.3 Element requirements**

2181 **7.3.27.3.1 General**

2182 Table 41 lists the element requirements for the CollectionSubscription adaptation.

2183 **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: *

Elements	Requirement	Description
Handler	Mandatory	Key: Value shall reference the ListenerDestination instance Multiplicity: *
Operations		
CreateInstance()	Mandatory	See 7.3.27.3.2 and DSP0223 .

2184 **7.3.27.3.2 Operation: CreateInstance()**

2185 Table 42 lists the error reporting requirements for the CreateInstance() operation on
 2186 CollectionSubscription instances. If any of the error situations described in the Description column of
 2187 Table 42 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 2188 addition, the error reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

2189 **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.		

2190 If the CreateInstance() operations is successful, the requested filter subscription was created, and
 2191 consequently — as required by 7.3.27.4 — shall be represented by a CollectionSubscription instance in
 2192 the requested namespace. In addition, if the requested namespace is not the Interop namespace, the
 2193 implementation shall expose an additional CollectionSubscription instance representing the subscription
 2194 in the Interop namespace (see 7.3.27.4).

2195 If the CreateInstance() operation fails, no subscription shall be created, and — as a consequence — no
 2196 representing CollectionSubscription instances shall be exposed in any namespace.

2197 Clients should abstain from requesting the creation of CollectionSubscription instances in namespaces
 2198 other than the Interop namespace.

2199 **7.3.27.4 Instance requirements**

2200 The instance requirements of 7.3.25.4 apply respectively for CollectionSubscription instances.

2201 **DEPRECATED**

2202 **7.3.28 ProfileOfFilterCollection: CIM_ConcreteDependency**

2203 The ProfileOfFilterCollection adaptation models the relationship between a filter collection defined in a
 2204 referencing profile and the profile registration of that referencing profile.

2205 The implementation type of the ProfileOfFilterCollection association adaptation is: "instantiated".

2206 Each StaticFilterCollection instance (see 7.3.17) representing a filter collection defined in a referencing
 2207 profile shall be associated through a ProfileOfFilterCollection instance with the ProfileRegistration
 2208 instance (see [DSP1033](#)) representing the implemented version of the referencing profile.

2209 NOTE This profile assumes that a future version of the Profile Registration profile (see [DSP1033](#)) will be based
 2210 on version 1.1 of the Profile Usage Guide (see [DSP1001](#)), and define the ProfileRegistration adaptation;
 2211 until then, substitute that by the definition of the CIM_RegisteredProfile "profile class" defined in version
 2212 1.0 of [DSP1033](#).

2213 Table 43 lists the element requirements for the ProfileOfFilterCollection adaptation.

2214 **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 .
EnumerateInstances()	Mandatory	See DSP0223 .
EnumerateInstanceNames()	Mandatory	See DSP0223 .

2215 **DEPRECATED**

2216 **7.3.29 BasicIndication: CIM_Indication**

2217 **7.3.29.1 General**

2218 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2219 The BasicIndication adaptation models indications; indications are described in 6.1.

2220 The implementation type of the BasicIndication indication adaptation is: "abstract".

2221 **7.3.29.2 Event definition requirements**

2222 Referencing profiles that model indications through adaptations based on the BasicIndication adaptation
 2223 shall define event that the indication is designed to report. This event definition shall be accomplished by
 2224 means of an event definition query statement stated in CQL (see [DSP0202](#)).

2225 The purpose of an event definition query statement is to formally define the event(s) that an indication
 2226 adaptation is designed to report, such that by inspecting the event definition query statements an

- 2227 implementer knows how to implement the indication adaptation. A CIM representation of event definition
 2228 query statements is not defined, thus there is no requirement for implementations or clients to be able to
 2229 programmatically interpret event definition query statements.
- 2230 NOTE Event definition query statements are different from indication filter query statements. An indication filter
 2231 query statement (see 7.3.11.3.5) defines the coverage of an indication filter, and is exposed to clients by
 2232 the value of the Query property in the IndicationFilter instance representing the indication filter. The
 2233 IndicationSpecificIndicationFilter adaptation (see 7.3.15) models indication-specific indication filters (see
 2234 6.2.4) and addresses the needs of clients requiring notifications about events reported by particular
 2235 indications specified in a profile.
- 2236 The CQL query statement defining the event shall comply with the following ABNF rule:
- 2237 "SELECT" WS PropertySet WS "FROM" WS IndicationClass WS "WHERE" WS
 2238 SelectionExpression
- 2239 PropertySet shall be "*", or a comma-separated list of property names.
- 2240 IndicationClass shall be the adapted indication class, that is, CIM_Indication or a subclass thereof.
- 2241 SelectionExpression shall be a constant string that defines a selection expression conformant with
 2242 the rules for selection expressions defined by [DSP0202](#).
- 2243 WS represents one or more whitespace characters.
- 2244 The requirements in this subclause may be refined by requirements defined in adaptations based on the
 2245 BasicIndication adaptation, including the case that a refined query statement references an external
 2246 element (such as an alert message definition in a message registry) that defines the event.
- 2247 **7.3.29.3 Indication origin**
- 2248 Each indication shall be assigned an origin namespace (see 6.1.2.4).
- 2249 In general, an implementation is free to select any local namespace as the origin namespace for a
 2250 generated indication; however, adaptations based on the BasicIndication adaptation such as the
 2251 AlertIndication adaptation (see 7.3.31) and the LifecycleIndication (see 7.3.32) establish additional
 2252 constraints.
- 2253 The indication origin is not represented in the CIM representation of an indication as defined by the
 2254 CIM_Indication class.
- 2255 The implementation class of the indication is required to reside in the origin namespace.
- 2256 NOTE As with any implementation class, the existence of an indication implementation class within a namespace
 2257 is does not sufficiently indicate that the indication is really implemented. Additional requirements — such
 2258 as the presence and integration of functional code implementing the indication — apply, but are outside of
 2259 the scope of this profile.
- 2260 The indication origin is required to be considered during indication filtering; see 6.1.4 and 7.3.11.2.
- 2261 **7.3.29.4 Element requirements**
- 2262 **7.3.29.4.1 General**
- 2263 Table 44 lists the element requirements for the BasicIndication adaptation.

2264

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.

2265 **7.3.29.4.2 Property: IndicationFilterName**

2266 The value of the IndicationFilterName property shall contain the name of the indication gate that the
 2267 indication passed before being delivered to the listeners subscribed to that indication gate. For indication
 2268 filters, the name is exposed by the value of the Name property in representing IndicationFilter instances
 2269 (see 7.3.11). For filter collections, the name is exposed by the value of the CollectionName property in
 2270 representing StaticFilterCollection instances (see 7.3.17).

2271 Because an indication is generated independently and before it is subjected to filtering, the name of the
 2272 filtering indication gate is not known at indication-generation time. Instead, a generated indication might
 2273 match a large number of indication gates. During indication filtering (see 6.1.4 and 7.3.11.2), each time a
 2274 generated indication matches an indication gate with existing subscriptions, and before delivering that
 2275 indication to subscribed listeners, the implementation shall set the value of the IndicationFilterName
 2276 property in the BasicIndication instance representing the indication to the identification of that indication
 2277 gate, as follows:

- 2278 • in case of indication filters, the identification shall be the value of the Name property of the
 2279 IndicationFilter instance representing the indication filter
- 2280 • in case of filter collections, the identification shall be the value of the CollectionName property of
 2281 the StaticFilterCollection instance representing the filter collection.

2282 NOTE 1 The requirement for referencing filter collections was added with version 1.2. of this profile.

2283 NOTE 2 A listener may use the value of the IndicationFilterName property to determine which indication gate was
 2284 passed by the indication before being delivered to the listener.

2285 **7.3.29.5 Indication generation requirements**

2286 Adaptations based on the BasicIndication adaptation are required to define the event that the modeled
 2287 indication is designed to report; see 7.3.29.2.

2288 If the event defined by such an adaptation occurs, and if subscriptions exist for any indication gate
 2289 covering the modeled indication, an instance of the indication adaptation based on the BasicIndication
 2290 shall be generated.

2291 NOTE The way this requirement is stated it provides for the optimized approach of checking for the presence of
 2292 matching indication gate with subscriptions already at indication generation time; however, even in this
 2293 case indication filtering is required as a subsequent step (see 6.1.4) in order to ensure that all matching
 2294 indication gates are considered, and indication delivery occurs to all listeners subscribed to any of the
 2295 indication gates covering the indication.

2296 **7.3.30 ReliableIndication: CIM_Indication**

2297 **7.3.30.1 General**

2298 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2299 The ReliableIndication adaptation models reliable indications; the concept of reliable indications is
 2300 introduced in 6.1.5. Additional requirements for reliable indication delivery are specified in 7.4.

2301 The implementation type of the ReliableIndication indication adaptation is: "abstract".

2302 NOTE The ReliableIndications adaptation is intentionally not based on the BasicIndication adaptation, such that it
 2303 can be implemented independently as a separate option. Reliable indication delivery is typically
 2304 implemented centrally once for the delivery of all indications implemented by an implementation.

2305 **7.3.30.2 Element requirements**

2306 **7.3.30.2.1 General**

2307 Table 45 lists the element requirements for the ReliableIndication adaptation.

2308 **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.

2309 **7.3.30.2.2 Property: SequenceContext**

2310 The value of the SequenceContext property shall contain the sequence context portion of the sequence
 2311 identifier (see 3.30 and 7.4.2). See the CIM schema description for additional constraints and the required
 2312 semantics, and see 7.4 for additional requirements on reliable indication delivery.

2313 NOTE 1 The CIM schema definition of the CIM_Indication class requires for the SequenceContext property that the
 2314 implementation maintains the context for this property separately for each registered listener destination,
 2315 and that restarts of the WBEM server cause the value to change. This requirement enables a listener to
 2316 detect WBEM server restarts, and to differentiate the indication streams from a particular WBEM server
 2317 that were processed (within that WBEM server) through different listener destinations referring to the
 2318 listener.

2319 NOTE 2 Indications can be lost when a listener fails and restarts, with the WBEM server continuing to send
 2320 indications while the listener is inactive. In that case, upon restart of the listener, if does not persist the last
 2321 received sequence identifier, the listener would establish the sequence identifier of the first received
 2322 indication after the restart as check value, failing to notice that while it was inactive additional indications
 2323 were sent (and lost). One approach for discovering an actual loss of indications might be to persist the
 2324 latest sequence identifier as part of a listener termination routine, and upon restart use the persisted value
 2325 as a check value (instead of that taken from the first arriving indication after the restart).

2326 **7.3.30.2.3 Property: SequenceNumber**

2327 The value of the SequenceNumber property shall contain the sequence number portion of the sequence
 2328 identifier (see 3.30 and 7.4.2). See the CIM schema description for additional constraints and the required
 2329 semantics, and see 7.4 for additional requirements on reliable indication delivery.

2330 NOTE The CIM schema definition of CIM_Indication class requires for the SequenceNumber property in the
 2331 stream of instances processed through a particular listener destination, that the value starts at 0 whenever
 2332 the value of the SequenceContext property changes.

2333 **7.3.31 AlertIndication: CIM_AlertIndication**2334 **7.3.31.1 General**

2335 The AlertIndication adaptation models alert indications; alert indications are described in 6.1.3.

2336 The implementation type of the AlertIndication indication adaptation is: "abstract".

2337 It is expected that the AlertIndication adaptation is used as a base adaptation for modeling alert
2338 indications in referencing profiles.

2339 **7.3.31.2 Event definition requirements**

2340 This subclause refines the event definition requirements established by the BasicIndication adaptation;
2341 see 7.3.29.2.

2342 The query statement defined by the following ABNF rules define the event(s) that are reported by
2343 AlertIndication instances:

- 2344 • If the AlertIndication adaptation identifies only one related alert message (see 7.3.31.3), the
2345 event query statement is defined as follows:

```
2346 EventQuerySingle = "SELECT" WS PropertySet WS "FROM" WS
2347 AlertIndicationClass WS "WHERE" WS "OwningEntity='" OwningEntity "'"
2348 WS "AND" WS "MessageID=" MessageId WS AdditionalWhereElements
```

- 2349 • If the AlertIndication adaptation identifies more than one related alert message (see 7.3.31.3),
2350 the event query statement is defined as follows:

```
2351 EventQueryMulti = "SELECT" WS PropertySet WS "FROM" WS
2352 AlertIndicationClass WS "WHERE" WS "OwningEntity='" OwningEntity "'"
2353 WS "AND" WS "MessageID LIKE" WS "'" MessageSet "'" [ WS
2354 AdditionalSelectionExpression ]
```

```
2355 MessageSet = MessageIdentification [ "|" MessageSet ]
```

2356 NOTE Recall that the purpose of the event definition query statement is to formally define the event(s) that an
2357 indication is designed to report; see 7.3.29.2. Event definition query statements are not represented in
2358 CIM; thus there is no requirement for implementations or clients to interpret event definition query
2359 statements.

2360 `PropertySet` shall be "*", or a comma-separated list of property names.

2361 `AlertIndicationClass` shall be `CIM_AlertIndication`, or, if adaptations based on the
2362 `AlertIndication` adaptation adapt a class derived from `CIM_AlertIndication`, shall be replaced by the name
2363 of the adapted alert indication class.

2364 `OwningEntity` shall be the name of the organization defining the alert indication. In profiles owned by
2365 DMTF, the value shall be "DMTF".

2366 `MessageIdentification` shall identify each referenced alert message, as required by 7.3.31.3.

2367 Referencing profiles in their adaptations based on the `AlertIndication` adaptation may refine the event
2368 definition; however, such refinements shall remain within the constraints established by the query
2369 statement specified in this subclause.

2370 If a referencing profile defining an adaptation based on the `AlertIndication` adaptation does not require
2371 refining the query statement specified in this subclause, then a repetition of the query statement is not
2372 required as part of the adaptation in the referencing profile, and compliance with this subclause is
2373 achieved through designating a related alert message as required in 7.3.31.3.

2374 `AdditionalSelectionExpression` shall be a constant string that defines a selection expression
2375 conformant with the rules for selection expressions defined by [DSP0202](#). For example, the value of the
2376 `PerceivedSeverity` property could be constrained to specific values.

2377 7.3.31.3 Related alert messages

2378 Referencing profiles defining adaptations based on the `AlertIndication` adaptation as part of their alert
2379 indication adaptation shall reference one or more related CIM alert message(s) that are defined in a
2380 message registry conformant to [DSP0228](#).

2381 The formal requirements for referencing alert messages through message identifications as part of
2382 adaptation definitions are detailed in [DSP1001](#); as defined there, the main elements of a message
2383 identification are the name of the registry reference referring to the registry defining the alert message,
2384 and the message id as the concatenation of the value of the `PREFIX` attribute and the
2385 `SEQUENCE_NUMBER` attribute from the `MESSAGE_ID` element that defines the message within the
2386 message registry.

2387 CIM alert messages provide for a formalized and widely self-contained approach to define alert
2388 indications. CIM alert messages are defined in message registries. A message registry is an XML
2389 document that contains message definitions. [DSP0228](#) defines an XML schema for message registries.
2390 The schema defines the XML elements that can be used for message definitions. Each element is
2391 formally defined using the XML schema language. Each of these element definitions is annotated with
2392 documentation that may define formal requirements for the use of the message element.

2393 Each message definition in a message registry consists of a standard message identifier and a
2394 description of static and dynamic message elements and of other message components; for details, see
2395 [DSP0228](#).

2396 The `MESSAGE_ID` element within the message definition identifies the message within the scope of the
2397 message registry through a prefix and a sequence number.

2398 The `MESSAGE_DESCRIPTION` element within an alert message definition contains a plain text description
2399 of the event that is reported by the defined alert message. A profile modeling an alert indication shall rely
2400 on the event definition provided in the alert message description. In case the alert-message-based
2401 definition of the event is insufficient in the context of the profile, the profile may augment the event
2402 definition within its definition of the alert indication; however, the amendments to the event definition
2403 stated in a profile shall remain within the constraints defined by the event definition in the alert message
2404 definition in the message repository.

2405 The `<MESSAGE_COMPONENTS>` element within an alert message definition defines a sequence of static
2406 and dynamic elements that together compose the message. The static elements define constant text
2407 parts of the message. The dynamic elements reference property values in identified CIM instances, such
2408 that the property values become dynamic parts of the alert message.

2409 7.3.31.4 Indication origin

2410 If the alert indication is related to a managed object, and the CIM representation of that managed object is
2411 referenced by the value of the `AlertingManagedElement` property in the CIM representation of the alert
2412 indication, then the indication origin as required by 7.3.29.3 should be the namespace in which the CIM
2413 representation of that managed object exists.

2414 7.3.31.5 Element requirements

2415 7.3.31.5.1 General

2416 Table 46 lists the element requirements for the `AlertIndication` adaptation.

2417

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.

2418 7.3.31.5.2 Property: AlertingManagedElement

2419 If the managed element for which the alert indication is reported is represented by one or more CIM
 2420 instances within the implementation, then the value of the AlertingManagedElement property shall identify
 2421 the most prominent of these CIM instances, using the format of a WBEM-URI-UntypedInstancePath (as
 2422 defined in [DSP0207](#)); otherwise the value of the AlertingManagedElement property shall be Null.

2423 7.3.31.5.3 Property: AlertType

2424 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the AlertType property in
 2425 CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2426 content of the ALERT_TYPE element from the alert message definition in the message registry.

2427 7.3.31.5.4 Property: Message

2428 The requirement level of the Message property is optional.

2429 The Message property may contain the formatted alert message from the registry.

2430 7.3.31.5.5 Property: MessageID

2431 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the MessageID property in
 2432 CIM_AlertIndication instances conveying an alert message from a message registry to be set to the

2433 concatenation of the `PREFIX` and `SEQUENCE_NUMBER` attribute values from the alert message definition
 2434 in the message registry (that is, no further padding or adjustment of these values takes place).

2435 NOTE The `SEQUENCE_NUMBER` attribute value is not to be confused with the sequence number within a
 2436 sequence identifier that enables unique identification of the indications originating from a particular WBEM
 2437 server to a particular WBEM listener; see 7.4.2.

2438 7.3.31.5.6 Property: OwningEntity

2439 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the `OwningEntity` property
 2440 in `CIM_AlertIndication` instances conveying an alert message from a message registry to be set to the
 2441 content of the `OWNING_ENTITY` element from the alert message definition in the message registry.

2442 7.3.31.5.7 Property: PerceivedSeverity

2443 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the `PerceivedSeverity`
 2444 property in `CIM_AlertIndication` instances conveying an alert message from a message registry to be set
 2445 to the content of the `PERCEIVED_SEVERITY` element from the alert message definition in the message
 2446 registry.

2447 7.3.31.5.8 Property: SystemName

2448 If the managed element for which the alert indication is reported is represented by a CIM instance within
 2449 the implementation, and the managed element is a component of a system that is represented by a
 2450 `CIM_System` instance, then the value of the `SystemName` property in the `AlertIndication` instance shall be
 2451 identical with the value of the `Name` property in the `CIM_System` instance; otherwise, the value of the
 2452 `SystemName` property shall be Null.

2453 7.3.31.5.9 Property: MessageArguments[]

2454 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the (string typed) `MessageArguments`
 2455 array property in `CIM_AlertIndication` instances conveying an alert message from a message registry to
 2456 contain one array entry for each dynamic element defined in the alert message, in the order specified by
 2457 the alert message definition in the message registry, where the value of the array element provides the
 2458 value of the dynamic element.

2459 If for a particular alert indication defined by a referencing profile the definition of a dynamic element
 2460 (including its description) within an alert message definition in a message registry is not sufficient to
 2461 identify a particular CIM instance and property as required by the referencing profile, then the referencing
 2462 profile shall specify augmenting provisions that explicitly identify an instance and a property that are
 2463 compatible with the definition of the dynamic element within the alert message.

2464 For example, assume that an alert message is defined in a message repository, as follows:

```

2465 <MESSAGE NAME="System state change">
2466   <MESSAGE_ID PREFIX="SVPC" SEQUENCE_NUMBER="0123"/>
2467   <MESSAGE_DESCRIPTION>
2468     This message describes a system state change.
2469   </MESSAGE_DESCRIPTION>
2470   <MESSAGE_COMPONENTS>
2471     <STATIC_ELEMENT>The system </STATIC_ELEMENT>
2472     <DYNAMIC_ELEMENT NAME="SystemElementName"
2473       SOURCE_PROPERTY="CIM_System.ElementName" DATATYPE="string"/>
2474     <STATIC_ELEMENT> changed its state to </STATIC_ELEMENT>
2475     <DYNAMIC_ELEMENT NAME="SystemState"
2476       SOURCE_PROPERTY="CIM_System.EnabledState" DATATYPE="string"/>
2477     <STATIC_ELEMENT> .</STATIC_ELEMENT>
2478   </MESSAGE_COMPONENTS>
2479   <FIXED_MESSAGE_INSTANCE_VALUES TYPE="ALERT">
```

```

2480     <!-- . . . -->
2481     </FIXED_MESSAGE_INSTANCE_VALUES>
2482     <!-- . . . -->
2483     </MESSAGE>

```

2484 An Example System Virtualization profile might model an indication reporting state changes of both host
 2485 systems and virtual systems. In both cases the SVPC0123 alert message would be used, but the
 2486 identification of affected instances would need to be specialized separately for each case.

2487 Assuming that the profile defines a HostSystem adaptation of the CIM_System class for the
 2488 representation of host systems, and defines a HostStateChange indication adaptation in order to report
 2489 state changes of host systems, the requirements for the MessageArguments[] array property as part of
 2490 the HostStateChange indication adaptation would need to augment the alert message definition from the
 2491 message registry, as follows:

- 2492 • The value of MessageArguments[0] shall be the value of the ElementName property of the
 2493 HostSystem instance representing the host system that changed its state.
- 2494 • The value of MessageArguments[1] shall be the new value of the EnabledState property of the
 2495 HostSystem instance representing the host system that changed its state.

2496 7.3.31.6 Indication generation requirements

2497 The indication generation requirements of 7.3.29.5 apply respectively for the AlertIndication adaptation.

2498 7.3.32 LifecycleIndication: CIM_InstIndication

2499 7.3.32.1 General

2500 The LifecycleIndication adaptation models lifecycle indications of CIM instances; lifecycle indications are
 2501 described in 6.1.2.3.

2502 The LifecycleIndication adaptation adapts the CIM_InstIndication class and is based on the
 2503 BasicIndication adaptation (see 7.3.29); in addition, if the ReliableIndications feature (see 7.2.4) is
 2504 implemented, it is also based on the ReliableIndication adaptation (see 7.3.30).

2505 The implementation type of the LifecycleIndication indication adaptation is: "abstract".

2506 It is expected that the LifecycleIndication adaptation is used as a base adaptation for modeling lifecycle
 2507 indications in referencing profiles.

2508 7.3.32.2 Event definition requirements

2509 This subclause refines the event definition requirements established by the BasicIndication adaptation
 2510 (see 7.3.29.2) for the LifecycleIndication adaptation.

2511 Recall that lifecycle indication reports secondary events (see 6.1.1). The secondary event that is reported
 2512 by LifecycleIndication instances shall be described by an event definition query statement that conforms
 2513 to the following ABNF rule:

```

2514     "SELECT" WS PropertySet WS "FROM" WS LifecycleIndicationClass WS
2515     "WHERE" WS "ISA" WS ModelElement [ WS "WHERE" SelectionExpression ]

```

2516 PropertySet shall be "*", or a comma-separated list of property names.

2517 LifecycleIndicationClass shall be one of CIM_InstCreation, CIM_InstDeletion, or
 2518 CIM_InstModification, or a subclass of these indication classes.

2519 ModelElement shall identify a class for that the referencing profile defines a class adaptation, and for
 2520 which the modeled lifecycle indication reports secondary events. The class adaptation of that class shall
 2521 be stated as part of the description of the lifecycle indication adaptation in the referencing profile.

2522 NOTE For examples that comply with this requirement, see 7.3.33 and 7.3.34.

2523 SelectionExpression shall be a constant string that defines a selection expression conformant with
 2524 the rules for selection expressions defined by [DSP0202](#).

2525 NOTE These rules provide for referencing profiles being able to define one lifecycle indication for one target
 2526 adaptation per lifecycle indication adaptation. If for a particular target adaption a referencing profile intends
 2527 to model lifecycle indications for different lifecycle events (such as the creation, destruction or modification
 2528 of instances of the target adaptation), for each of these lifecycle events separate lifecycle indication
 2529 adaptations are required. Furthermore, if lifecycle indications are to be modeled for different target
 2530 adaptations, for each target adaptation separate lifecycle indication adaptations are required. As usual, if
 2531 common requirements exist for such lifecycle indication adaptations, these can be defined in a common
 2532 abstract base adaptation that is used as a base for the specific lifecycle indication adaptations, thereby
 2533 avoiding the repetition of the commonalities.

2534 **7.3.32.3 Indication origin**

2535 The indication origin as required by 7.3.29.3 shall be the namespace of the CIM instance referenced by
 2536 the value of the SourceInstanceModelPath property (see 7.3.32.4.3).

2537 **7.3.32.4 Element requirements**

2538 **7.3.32.4.1 General**

2539 Table 47 lists the element requirements for the LifecycleIndication adaptation.

2540 **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.

2541 **7.3.32.4.2 Property: SourceInstance**

2542 The value of the SourceInstance property shall be an embedded instance of the class selected in the
 2543 query statement defining the event. The embedded instance shall be a copy of the instance for which the
 2544 lifecycle indication is reported. If the query statement specifies a specific selection of properties (other
 2545 than "*"), then the set of properties contained in the embedded instance shall be limited to those
 2546 selected; otherwise, the embedded instance shall at least contain values for each of the properties
 2547 required by the related adaptation of the selected class in the same referencing profile; see 7.3.29.2.

2548 **7.3.32.4.3 Property: SourceInstanceModelPath**

2549 The value of the SourceInstanceModelPath property shall refer to the same instance that is copied as an
 2550 embedded instance through the value of the SourceInstance property.

2551 **7.3.32.5 Indication generation requirements**

2552 The indication generation requirements of 7.3.29.5 apply respectively for the LifecycleIndication
2553 adaptation.

2554 **7.3.33 ListenerDestinationRemovalIndication: CIM_InstDeletion**

2555 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2556 The ListenerDestinationRemovalIndication adaptation models a lifecycle indication that reports the
2557 destruction of a CIM_ListenerDestination instance, as modeled in this profile by the ListenerDestination
2558 adaptation (see 7.3.23). The destruction of a ListenerDestination instance is a secondary event caused
2559 by the destruction of the represented listener destination; see 6.4.5.

2560 The requirement level of the ListenerDestinationRemovalIndication indication adaptation is optional.

2561 The implementation type of the ListenerDestinationRemovalIndication indication adaptation is:
2562 "indication".

2563 Table 48 lists the element requirements for the ListenerDestinationRemovalIndication adaptation.

2564 **Table 48 – ListenerDestinationRemovalIndication: Element requirements**

Elements	Requirement	Description
Base adaptations		
LifecycleIndication	Mandatory	See 7.3.32.

2565 The requirement level of the ListenerDestinationRemovalIndication adaptation is optional.

2566 The event reported by the ListenerDestinationRemovalIndication adaptation is defined by the following
2567 event definition query statement:

```
2568     SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA
2569     CIM_ListenerDestination
```

2570 **7.3.34 SubscriptionRemovalIndication: CIM_InstDeletion**

2571 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2572 The SubscriptionRemovalIndication adaptation models a lifecycle indication that reports the destruction of
2573 a CIM_AbstractIndicationSubscription instance, as modeled in this profile by the AbstractSubscription
2574 adaptation (see 7.3.25). The destruction of a CIM_AbstractIndicationSubscription instance is a secondary
2575 event caused by the destruction of the represented subscription; see 6.1.1.

2576 The requirement level of the SubscriptionRemovalIndication indication adaptation is optional.

2577 The implementation type of the SubscriptionRemovalIndication indication adaptation is: "indication".

2578 Table 49 lists the element requirements for the SubscriptionRemovalIndication adaptation.

2579 **Table 49 – SubscriptionRemovalIndication: Element requirements**

Elements	Requirement	Description
Base adaptations		
LifecycleIndication	Mandatory	See 7.3.32.

2580 The requirement level of the SubscriptionRemovalIndication adaptation is optional.

2581 The event reported by the SubscriptionRemovalIndication adaptation is defined by the following query
2582 statement:

```
2583     SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA  
2584     CIM_AbstractIndicationSubscription
```

2585 **7.4 Reliable indication delivery**

2586 **7.4.1 General**

2587 This subclause defines mechanisms for the reliable delivery of indications from an implementation to a
2588 listener as described in 6.1.5.

2589 Implementations implementing the ReliableIndications feature (see 7.2.4) shall comply with the
2590 requirements specified in 7.4.3; note that in addition the requirements of the ReliableIndications
2591 adaptation (see 7.3.30) apply.

2592 Implementations not implementing the ReliableIndications feature are not required to comply with the
2593 provisions in this subclause or those in 7.3.30.

2594 Listeners implementing the ReliableIndications feature (see 7.2.4) shall comply with the provisions stated
2595 in 7.4.4. Listeners not implementing the ReliableIndications feature are not required to comply with these
2596 provisions and may ignore the sequence identifiers in received indications, as exposed by the values of
2597 the SequenceContext and SequenceNumber properties in any received CIM_Indication instances.

2598 **7.4.2 Sequence identifier and sequence identifier lifetime**

2599 This subclause defines the concepts of *sequence identifier* and *sequence identifier lifetime*.

2600 The *sequence identifier* within an indication enables unique identification of the indications originating
2601 from a particular WBEM server to a particular WBEM listener.

2602 A sequence identifier is composed of a sequence context and a sequence number.

2603 NOTE The sequence number within a sequence identifier is not to be confused with the `SEQUENCE_NUMBER`
2604 attribute value that is part of the identification of the alert message that defines an alert indication; see
2605 7.3.31.5.5.

2606 The sequence context is required to be unique for each listener destination maintained by the indication
2607 service within a WBEM server; within that context the sequence number is required to be unique for each
2608 indication delivered from the WBEM server to the listener referenced by the listener destination. The
2609 requirements for the CIM representation of the sequence identifier in reliable indications are defined in
2610 7.3.30.

2611 The *sequence identifier lifetime* maintained by an implementation is a duration defined as follows:

2612
$$\text{sequence-identifier-lifetime} = \text{number-of-retry-attempts} * \text{delivery-retry-interval} * 10$$

2613 In this formula the number-of-retry-attempts is the number of retry attempts as indicated by the value of
2614 the DeliveryRetryAttempts property (see 7.3.2.3.3) in the IndicationService instance representing the
2615 indication service within the implementation, and the delivery-retry-interval is the duration of the delivery
2616 retry interval as indicated by the value of the DeliveryRetryInterval property (see 7.3.2.3.4) in the same
2617 instance.

2618 Within the sequence identifier lifetime an implementation that is implementing reliable indications may
2619 attempt to retry failed indication delivery attempts, as detailed in 7.4.3, and a listener implementing
2620 reliable indications may expect the delivery of anticipated indications, as detailed in 7.4.4.

2621 7.4.3 WBEM server requirements

2622 7.4.3.1 General

2623 Indication delivery is based on a publish/subscribe event paradigm, where an implementation delivers
2624 indications to subscribed listeners. The indication delivery may fail for various reasons, including
2625 unavailability of the listener or network issues. This subclause describes the requirements for the
2626 implementation that are related to reliable indication delivery. The mechanisms to deliver indications and
2627 to determine success or failure of indication delivery are protocol dependent; see the specifications of
2628 applicable protocols that specify mechanisms for indication delivery.

2629 7.4.3.2 Prohibition of indication delivery for disabled or removed subscriptions

2630 If a subscription is disabled or has been removed, the implementation should discard any undelivered
2631 indications for that subscription. For example, this applies if the implementation has queued indications
2632 for delivery retry, and the subscription is removed by a client before the delivery retry is executed.

2633 7.4.3.3 Prohibition of repeated indication delivery

2634 After an implementation has successfully delivered an indication to a listener, it shall not deliver that
2635 indication again to that same listener.

2636 7.4.3.4 Requirements for the retry of failed indication deliveries

2637 If the attempt to deliver an indication to a particular listener fails, the implementation shall retry the
2638 indication delivery as detailed in this subclause.

- 2639 1) The implementation shall wait for the duration of the delivery retry interval, as exposed by the
2640 value of the `DeliveryRetryInterval` property in the `IndicationService` instance (see 7.3.2)
2641 representing the indication service within the implementation.
- 2642 2) If the actual number of retry attempts is less than the maximum number of retry attempts as
2643 exposed by the value of the `DeliveryRetryAttempts` property in the `IndicationService` instance
2644 representing the indication service within the implementation, and the elapsed time after the first
2645 delivery is less than the sequence identifier lifetime as defined in 7.4.2, the implementation shall
2646 retry the failed indication delivery.
 - 2647 • If the retry is successful, delivery of that indication to the particular listener is complete.
 - 2648 • If the retry is not successful, and preconditions of step 2) still apply, then the
2649 implementation shall re-iterate starting with step 1).
 - 2650 • Otherwise, the indication shall be considered as not deliverable to the particular listener,
2651 and the requirements defined in 7.4.3.5 apply.

2652 7.4.3.5 Requirements for undeliverable indications

2653 This subclause defines the implementation behavior if an indication has been considered unable to be
2654 delivered to a listener, as described in 7.4.3.4.

2655 If the listener destination referencing that listener is permanent (see 7.3.23.3.3), the implementation shall
2656 record an error and shall no longer attempt to deliver that indication to that listener (that is, the
2657 implementation shall discard it). This action does not modify the listener destination and any of its
2658 subscriptions.

2659 If the listener destination referencing that listener is transient (see 7.3.23.3.3), the implementation shall
2660 record an error and shall no longer attempt to deliver that indication to that listener (that is, the
2661 implementation shall discard it). In addition, the listener destination and its subscriptions may be removed
2662 from the implementation as described in 7.4.3.6.

2663 7.4.3.6 Requirements for the implicit removal of subscriptions and listener destinations

2664 An implementation may remove a subscription and the referenced listener destination if the delivery of
2665 one or more indications to the represented listener failed as described in 7.4.3.4 and 7.4.3.5.

2666 The implementation behavior with respect to the implicit removal of subscriptions and listener destinations
2667 shall be exposed by the value of the SubscriptionRemovalAction property in the IndicationService
2668 instance representing the responsible indication service; see 7.3.2.3.5.

2669 7.4.3.7 Behavior related to WBEM server restarts

2670 Indications that have been generated but not yet delivered may get lost during a WBEM server crash or
2671 restart because there is not requirement to persist such indications.

2672 If the implementation chooses an algorithm for the construction of the sequence context part of the
2673 sequence identifier (see 7.4.2) that includes the WBEM server startup time, the potential re-use of the
2674 same sequence identifier is implicitly avoided. That way listeners can deal with indication delivery failures
2675 caused by WBEM server restarts in the same way they deal with other kinds of indication delivery failures.

2676 7.4.4 WBEM listener requirements**2677 7.4.4.1 General**

2678 A listener shall keep track of each distinct sequence identifier of any indications received from a particular
2679 indication service for the duration of the sequence identifier lifetime maintained by that indication service,
2680 counting from the last time that sequence identifier was detected in a received indication from that
2681 indication service. If the same sequence identifier is used by two different indication services (for
2682 example, in two different implementations), the listener shall keep track of them independently.

2683 After the lifetime of a sequence identifier expires, the listener should discard the knowledge about that
2684 sequence identifier from that indication service. After the knowledge about a sequence identifier for an
2685 indication service has been discarded by the listener, a new usage of that sequence identifier in an
2686 indication from that indication service shall be treated by the listener like a new, unknown sequence
2687 identifier from that indication service.

2688 Keeping track of sequence identifiers in listeners enables the detection of lost and duplicate deliveries,
2689 and the detection and re-ordering of indications arriving out of order, as described in 7.4.4.5. Discarding
2690 the knowledge about sequence identifiers minimizes the resource requirements of the listener.

2691 7.4.4.2 Determination of the expected sequence identifier of the next indication

2692 From the sequence identifier of the last indication received from a particular implementation, a listener
2693 shall infer the expected sequence identifier of the next indication by incrementing the sequence number
2694 by 1, wrapping to an initial value of 0 if the maximum limit has been reached, and maintaining the
2695 sequence context.

2696 7.4.4.3 Lost indications

2697 If the sequence identifier of the next received indication sent from the same implementation does not
2698 match the expected value as described in 7.4.4.2, the listener shall consider the expected indication as a
2699 candidate for a lost indication. After waiting for the sequence identifier lifetime period as maintained by
2700 the implementation sending that indication, the listener shall conclude that the expected indication is lost.

2701 7.4.4.4 Duplicate indications

2702 Any additional indications received from the same implementation with the same sequence identifier shall
2703 be considered duplicates. In this case, the lifetime for the sequence identifier shall be adjusted starting

2704 with the delivery time of the most recently received duplicate indication, and adding the sequence
2705 identifier lifetime period as maintained by the implementation sending that indication.

2706 **7.4.4.5 Out-of-order indications**

2707 A listener that intends to re-establish the original order of indications before processing them needs to
2708 defer the processing of any prematurely arriving indication that does not have the expected sequence
2709 number, until the decision can be made as to whether the expected indications are lost.

2710 If the sequence identifier of the next received indication does not match the expected sequence identifier
2711 as described in 7.4.4.2, the listener shall cache such prematurely arriving indications and wait for delivery
2712 of the indication with the expected sequence identifier for a period of time defined by the sequence
2713 identifier lifetime (as defined in 7.4.4.1) of the last received indication from the same implementation.

2714 If the indication with the expected sequence identifier is not received during that period, the expected
2715 indication should be considered lost (see 7.4.4.3).

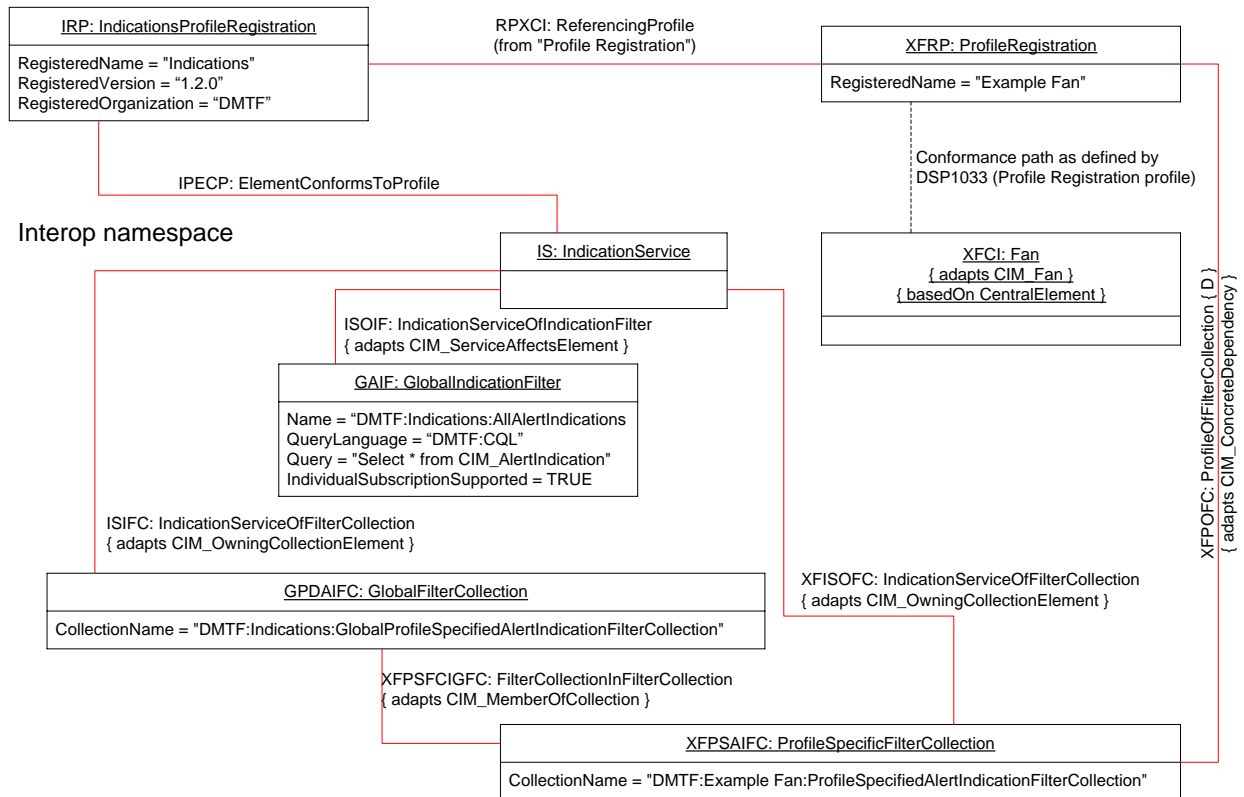
2716 If the indication with the expected sequence identifier is received during that period, the indication order
2717 shall be re-ordered using their sequence numbers, such that the indications are processed in the order
2718 they were sent by the implementation.

2719 **8 Use cases**

2720 **8.1 Object diagrams**

2721 Figure 4 depicts a DMTF object diagram. It shows CIM instances exposed by the implementation of an
 2722 Example Fan profile that defines some indications (not shown in the diagram), and thus is required by
 2723 [DSP1001](#) to reference this profile, implying the implementation of respective elements defined in this
 2724 profile.

2725



2726

2727 **Figure 4 – DMTF object diagram: Global and profile-specific filter collections**

2728 The implemented version of this profile is represented by the RegisteredProfile instance IRP, the
 2729 implemented version of the Example Fan profile is represented by RegisteredProfile instance XFRP, and
 2730 the reference relationship is shown by the ReferencingProfile association instance RPXCI.

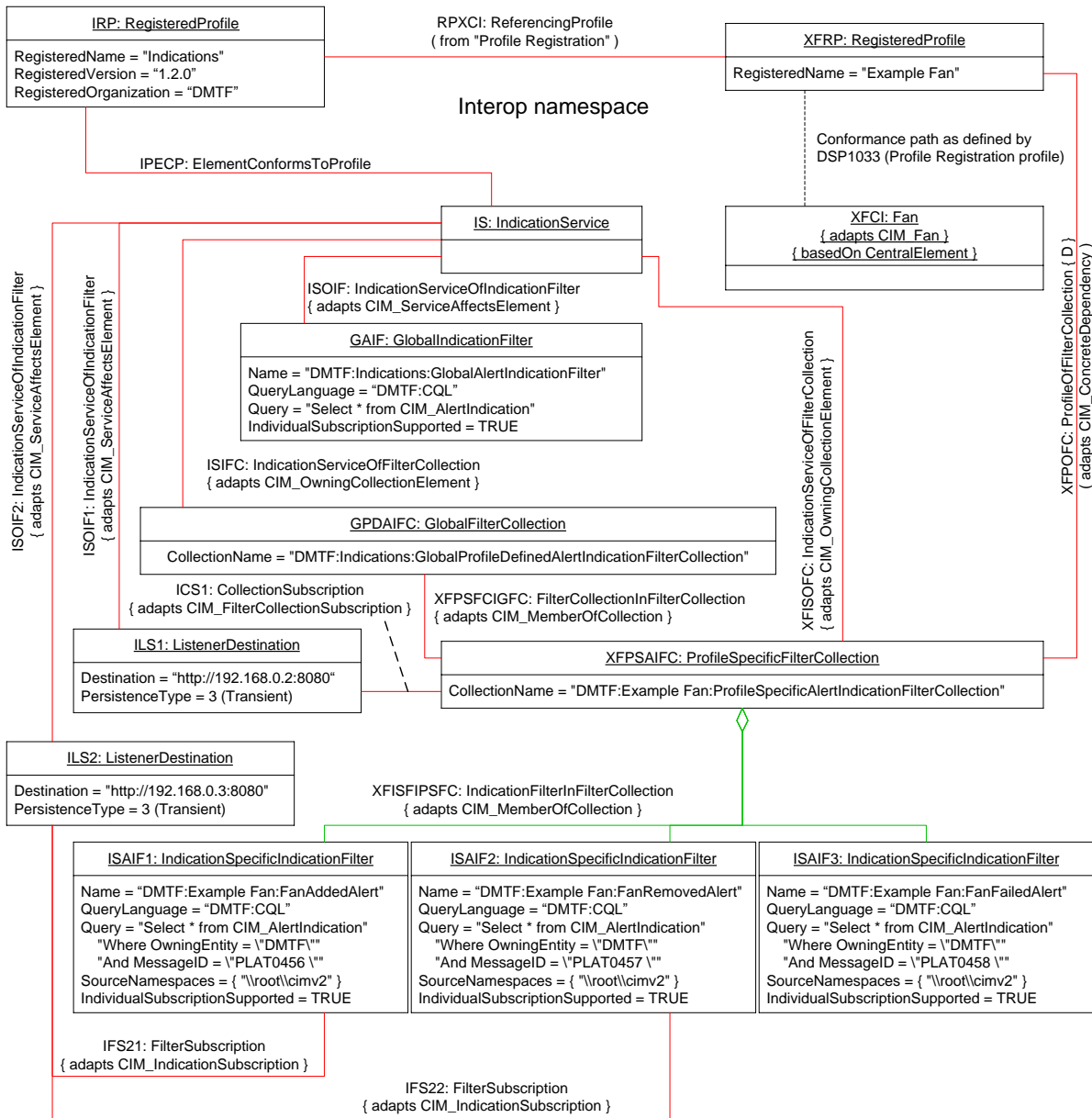
2731 The implementation of this profile exposes the IndicationService (see 7.3.2) instance IS representing the
 2732 implemented indication service. It also exposes the GlobalIndicationFilter (see 7.3.16) instance GAIF
 2733 representing the global indication filter covering all alert indications.

2734 Furthermore, the implementation of this profile exposes the GlobalFilterCollection (see 7.3.22) instance
 2735 GPDAIFC representing the global filter collection for alert indications with a defined coverage covering all
 2736 profile defined alert indications. The implementation of the Example Fan profile exposes the
 2737 ProfileSpecificFilterCollection (see 7.3.21) instance XFPSAIFC representing the related profile-specific
 2738 filter collection for alert indications with a defined coverage covering all alert indications defined in the
 2739 Example Fan profile.

2740 The global filter collection for alert indications represented by GPDAIFC contains the profile-specific filter
 2741 collection for alert indications represented by XFPSAIFC; this containment relationship is represented by
 2742 the FilterCollectionInFilterCollection (see 7.3.20) instance XFPSFCIGFC. Because the coverage of the

2743 global filter collection is explicitly represented by containment, in this case its coverage is inspectable by
2744 clients. However, the CIM representation of the contained profile-specific filter collection for alert
2745 indications represented by XFPSAIFC does not expose any contained elements. In that case clients
2746 would require prior knowledge of the defined coverage, that is, all alert indications defined in the Example
2747 Fan profile, which (because of the explicitly represented containment relationship) is in this example also
2748 the coverage of the global filter collection for alert indications represented by GPDAIFC.

2749 Figure 5 depicts a DMTF object diagram. It shows a variant of the situation illustrated in Figure 4.



root\cimv2 namespace

NOTE: The indications originate in this namespace, but do not exist in the namespace because they are transitional objects

XFALERT1: FanAddedAlert	XFALERT2: FanRemovedAlert	XFALERT3: FanFailedAlert
IndicationIdentifier = "XFALERT1" IndicationTime = "23:30:00 09/30/2009" OwningEntity = "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" OwningEntity = "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" OwningEntity = "DMTF" MessageID = "PLAT0458" AlertingManagedElement = "<URI referencing a CIM_Fan instance representing the failed fan->" AlertType = 5 (Device Alert) PerceivedSeverity = 4 (Minor)

2750

2751 **Figure 5 – DMTF object diagram: Filter collections and contained indication filters**

2752 The first difference from the situation shown in Figure 4 is that in Figure 5 the profile-specific filter
 2753 collection for alert indications represented by XFISAIFC contains three indication filters, represented by

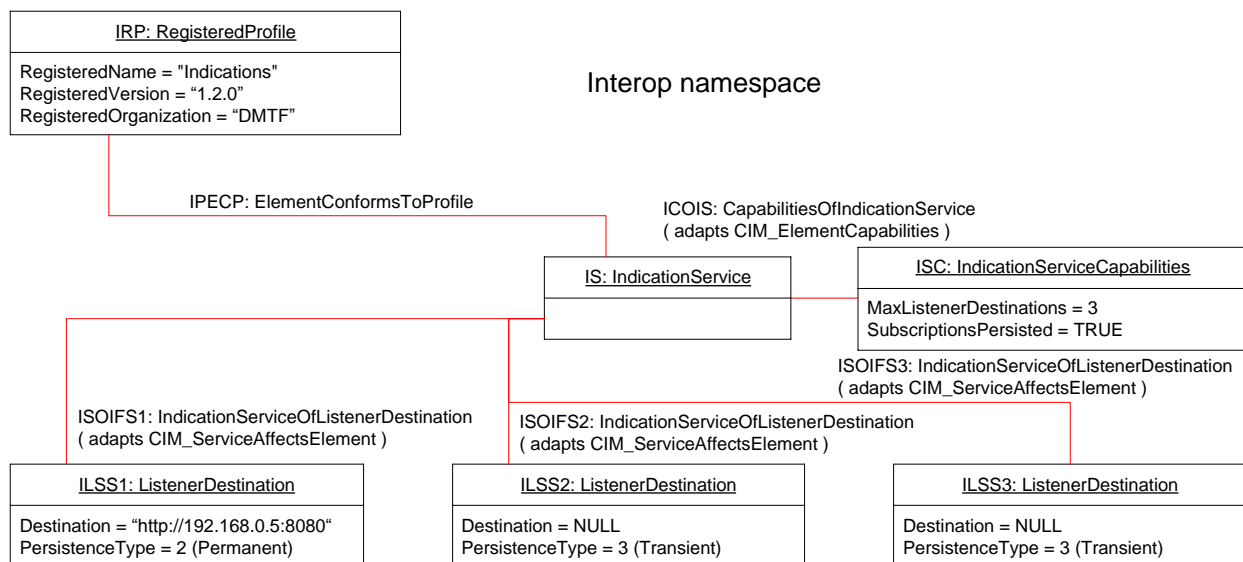
2754 the IndicationSpecificIndicationFilter instances ISAIF1, ISAIF2 and ISAIF3. Hence the coverage of the
 2755 profile-specific filter collection for alert indications represented by XFPSAIFC is now defined by the
 2756 contained indication filters, that is, it covers the three alert indications described by the alert messages
 2757 with the IDs PLAT0456, PLAT0457, and PLAT0458.

2758 It is important to recapture that — as with any indication gate — the presence of the CIM representation
 2759 of specific indication filters does not indicate that the covered indications are actually implemented. The
 2760 semantics of indication gates are defined with respect to *filtering*, but *not* with respect to *generating*,
 2761 indications (see 7.3.11.2 and 7.3.17.2). Thus, a subscribed listener is guaranteed only to be delivered any
 2762 *generated* indication that is within the coverage of the indication gate, but the *generation* of the indication
 2763 is not guaranteed. For that reason referencing profiles need to model other elements — such as
 2764 capabilities — for the purpose of conveying the information about which indications defined in the
 2765 referencing profile are actually implemented and thus generated when the respective event occurs; the
 2766 definition of such mechanisms is outside the scope of this profile.

2767 The second difference between Figure 4 and Figure 5 is that in Figure 5 listener destinations are
 2768 represented by the ListenerDestination instances ILS1 and ILS2. The listener referenced by ILS1 is
 2769 subscribed to the profile-specific filter collection represented by XFPSAIFC, and the listener referenced
 2770 by ILS1 is subscribed to the indication-specific indication filters represented by ISAIF1 and ISAIF2.

2771 Lastly, the representations of three indications are shown at the bottom of Figure 5, along with their origin
 2772 namespace. Each of these indications is within the coverage of the indication filter represented directly
 2773 above it. Thus, the alert indications represented by XFALERT1 and XFALERT2 are delivered to both the
 2774 listeners represented by ILS1 and ILS2, whereas XFALERT3 is only delivered to ILS1.

2775 Figure 6 depicts the DMTF object diagram for an implementation that supports a fixed number of listener
 2776 destinations.



2777
 2778

Figure 6 – DMTF object diagram: Static listener destinations

2779 In the example shown in Figure 6, an implementation supports a maximum of three listener destinations,
 2780 indicated by the value of the MaxListenerDestinations property in the IndicationServiceCapabilities
 2781 instance ISC that describes the capabilities of the indication service within the implementation. The three
 2782 listener destinations are represented by the three respective ListenerDestination instances ILSS1, ILSS2,
 2783 and ILSS3. The listener destination represented by ILSS1 is currently configured as a permanent listener
 2784 destination, referencing the listener reachable under URI "http://192.168.0.5:8080". The listener
 2785 destinations represented by ILSS2 and ILSS3 currently are free listener destinations as indicated by the
 2786 value Null for the Destination property, that is, they are not currently configured for a specific listener. A

2787 client can request modifications of any of the listener destinations in order to reference a desired listener
2788 for indication delivery by modifying the representing ListenerDestination instances.

2789 **8.2 LocateIndicationService: Locate the indication service provided by an** 2790 **implementation of this profile**

2791 **8.2.1 Preconditions**

2792 The client knows the following:

- 2793 • The identifying information of a WBEM server (for example, its IP address and the port number
2794 if the WBEM server implements CIM operations over http as described in [DSP0223](#))
- 2795 • Name, required version, and registered organization of this profile as stated in 7.3.5

2796 **8.2.2 Flow of activities**

2797 1) The client obtains all IndicationsProfileRegistration instances (see 7.3.5), applying respective
2798 use cases described in [DSP1033](#) to locate CIM_RegisteredProfile instances representing profile
2799 registrations of particular profiles and selecting those instances where the values of the
2800 RegisteredName, RegisteredVersion, and RegisteredOrganization properties match the
2801 required input values.

2802 The result is zero or more IndicationsProfileRegistration instances (see 7.3.23).

2803 NOTE 1 Typically only one instance is returned, but if this profile is implemented more than once within the
2804 identified WBEM server, more than one instance may be returned.

2805 If no instance was detected, this use case is complete and the client knows that the required
2806 version of this profile is not implemented within the WBEM server. If one or more instances
2807 were detected, any of them represents the required version of this profile, and the client can
2808 select any of these for further processing.

2809 2) The client applies use cases described in [DSP1033](#) in order to locate instances of the
2810 IndicationService adaptation that is the central class adaptation defined in this profile.

2811 The result is zero or one IndicationService instances (see 7.3.2).

2812 NOTE 2 Technically, more than one instance could be returned, but that would indicate a non-compliant
2813 implementation of this profile.

2814 If no instance was detected, this use case is complete and the client knows that an indication
2815 service is not presently active within the identified WBEM server. If one or more instances were
2816 detected, any of them represents an indication service compliant to the requirements specified
2817 in this profile, and the client can select any of these for further processing.

2818 **8.2.3 Postconditions**

2819 Unless errors occurred, the client either knows an IndicationService instance (including its object path)
2820 representing an indication service within the identified WBEM server with a behavior compliant to the
2821 requirements specified in this profile or knows that either this profile is not implemented within the
2822 identified WBEM server or that no indication service is presently active within the identified WBEM server.

2823 **8.3 LocateProfileIndicationService: Locate the indication service responsible for**
 2824 **delivering indications defined by a referencing profile**

2825 **8.3.1 Preconditions**

2826 The client knows the following:

- 2827 • The ProfileRegistration instance (including its object path) representing the profile registration of
 2828 the referencing profile

2829 **8.3.2 Flow of activities**

- 2830 1) For the input ProfileRegistration instance, find the IndicationsProfileRegistration instances (see
 2831 7.3.5) associated through ReferencedProfile instances (see [DSP1033](#)) (for example, using the
 2832 Associators() operation).

2833 The result is zero or one IndicationsProfileRegistration instances (see 7.3.5).

2834 NOTE 1 Technically, more than one instance could be returned, but that would indicate a non-compliant
 2835 implementation of the referencing profile.

2836 If no instance was detected, this use case is complete and the client knows that the
 2837 implementation of the referencing profile did not implement indications.

- 2838 2) For the IndicationsProfileRegistration instance obtained in step 1), find the IndicationService
 2839 instances (see 7.3.2) associated through ElementConformsToProfile instances (see 7.3.6) (for
 2840 example, using the Associators() operation).

2841 The result is zero or one IndicationService instances (see 7.3.2).

2842 NOTE 2 Technically, more than one instance could be returned, but that would indicate a non-compliant
 2843 implementation of this profile.

2844 **8.3.3 Postconditions**

2845 Unless errors occurred, the client knows an IndicationService instance (including its object path)
 2846 representing an indication service that is responsible for delivering indications defined by the referencing
 2847 profile.

2848 **8.4 DetermineIndicationServiceCapabilities: Determine the capabilities of an**
 2849 **indication service**

2850 **8.4.1 Preconditions**

2851 The client knows all of the following:

- 2852 • a copy of the IndicationService instance (including its object path) representing the indication
 2853 service within the implementation

2854 NOTE For example, that IndicationService instance could be obtained by applying the LocateIndicationService
 2855 use case (see 8.2) or the LocateProfileIndicationService use case (see 8.3).

2856 **8.4.2 Flow of activities**

2857 1) Inspecting property values of the IndicationService instance (see 7.3.2.3), the client can already
2858 determine some aspects of the behavior of the represented indication service.

2859 For example, the value of the FilterCreationEnabled property indicates whether the support for
2860 dynamic indication filters as modeled by the DynamicIndicationFilters feature (see 7.2.1) is
2861 available.

2862 The values of the DeliveryRetryAttempts, the DeliveryRetryInterval, the
2863 SubscriptionRemovalAction, and the SubscriptionRemovalTimeInterval indicate if and to what
2864 extent the support for reliable indications as modeled by the ReliableIndications feature (see
2865 7.2.4) is available.

2866 2) Find the IndicationsServiceCapabilities instance (see 7.3.7) representing the capabilities of the
2867 input indication service, by traversing the CIM_ServiceAffectsElement association modeled by
2868 the CapabilitiesOfIndicationService association adaptation (see 7.3.8) by invoking the
2869 Associators() operation with the following actual values for the input parameters:

- 2870 – InstanceName: the object path to the input IndicationService instance
- 2871 – AssocClass: "CIM_ElementCapabilities", the adapted class of the
2872 CapabilitiesOfIndicationService association adaptation
- 2873 – ResultClass: "CIM_IndicationServiceCapabilities", the adapted class of the
2874 IndicationServiceCapabilities adaptation

2875 The result is zero or one IndicationServiceCapabilities instance.

2876 NOTE Technically, more than one instance could be returned, but that would indicate a non-compliant
2877 implementation of this profile.

2878 If an IndicationServiceCapabilities instance was returned, the use case continues with step 3);
2879 otherwise, it continues with step 4).

2880 3) Inspect the property values of the returned IndicationServiceCapabilities instance (see 7.3.7).
2881 The values of those properties with names ending with "IsSettable" enable the client to
2882 determine whether client modification of respective aspects of the behavior of the input
2883 indication service is possible. The values of the MaxListenerDestinations and the
2884 MaxActiveSubscriptions properties expose the upper limits for the number of listener
2885 destinations and for the number of subscriptions supported by the indication service, and the
2886 value of the SubscriptionsPersisted property exposes whether subscriptions are persisted over
2887 restarts of the input indication service. This step completes this use case.

2888 4) Continue here after step 2) if no IndicationServiceCapabilities instance was returned. In this
2889 case, client modification of the indication service is not supported, and the upper limits for the
2890 number of supported listener destinations and subscriptions is not exposed by the
2891 implementation; in addition, whether subscriptions are persisted over indication service restarts
2892 is not exposed.

2893 **8.4.3 Postconditions**

2894 Unless errors occurred, the client knows the capabilities of the input indication service as far as it is
2895 exposed by the representing IndicationService instance, by the related IndicationServiceCapabilities
2896 instance, and by initial behavior specified in this profile.

2897 **8.5 ModifyIndicationService: Modify functional aspects of an indication service**

2898 The client knows all of the following:

- 2899 • a copy of the IndicationService instance (including its object path) (see 7.3.2) representing the
2900 indication service within the implementation (see the LocateIndicationService use case in 8.2)
- 2901 • a copy of the IndicationServiceCapabilities instance (including its object path) (see 7.3.7)
2902 representing the capabilities of the indication service within the implementation (See the
2903 DetermineIndicationServiceCapabilities use case in 8.4.)

2904 **8.5.1 Flow of activities**

- 2905 1) Inspect the property values in the input IndicationsServiceCapabilities instance (see 7.3.7)
2906 representing the capabilities of the input indication service to determine which properties in the
2907 IndicationService instance are modifiable. (See step 3) in the
2908 DetermineIndicationServiceCapabilities use case in 8.4.)
- 2909 2) If admissible by the determination of step 1), in the input local copy of the input
2910 IndicationService instance, modify property values as desired. For example, if the value of the
2911 DeliveryRetryAttemptsIsSettable property in the IndicationServiceCapabilities instance is True,
2912 a modification of the corresponding DeliveryRetryAttempts property in the IndicationService
2913 instance is admissible.
- 2914 3) Use the ModifyInstance() operation to request the desired change in the behavior of the
2915 indication service, providing the modified copy of the IndicationService instance as the actual
2916 value of the ModifiedInstance parameter.

2917 **8.5.2 Postconditions**

2918 Unless errors occurred, the desired change of functional aspects of the input indication service is
2919 effective.

2920 **8.6 ListListenerDestinations: List all listener destinations exposed by an** 2921 **implementation**

2922 **8.6.1 Preconditions**

2923 The client knows all of the following:

- 2924 • the object path to the IndicationService instance representing the indication service within the
2925 implementation (see 8.2)

2926 **8.6.2 Flow of activities**

- 2927 1) Find all listener destinations within the responsibility of the indication service by traversing the
2928 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfListenerDestination
2929 adaptation (see 7.3.24) by invoking the Associators() operation with the following actual values
2930 for the input parameters:
 - 2931 – InstanceName: the object path to the input IndicationService instance
 - 2932 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
2933 IndicationServiceOfListenerDestination adaptation
 - 2934 – ResultClass: "CIM_ListenerDestination", the adapted class of the ListenerDestination
2935 adaptation

2936 The result is a set of ListenerDestination instances (see 7.3.23).

2937 8.6.3 Postconditions

2938 Unless errors occurred, the client knows all ListenerDestination instances (including their object paths)
2939 representing all the listener destinations maintained by the implementation.

**2940 8.7 SelectListenerDestination: Select an existing listener destination referencing
2941 a desired listener****2942 8.7.1 Preconditions**

2943 The client knows all of the following:

- 2944 • the object path to the IndicationService instance representing the indication service within the
2945 implementation (see 8.2)
- 2946 • the URI exposed by the desired listener
- 2947 • the particular protocol to be applied when delivering these indications

2948 8.7.2 Flow of activities

2949 1) Execute the ListListenerDestinations use case (see 8.6).

2950 The result is a set of ListenerDestination instances (see 7.3.23).

2951 2) Inspect each ListenerDestination instance resulting from step 1) by checking the value of the
2952 Destination property against the input URI, and by checking whether the value of the Protocol
2953 property matches the particular protocol for this use case.

2954 If both conditions are met, the located ListenerDestination represents a listener destination that
2955 within the implementation represents the particular listener, and this use case is complete;
2956 otherwise, the client needs to repeat step 2), inspecting further ListenerDestination instances
2957 from the result of step 1).

2958 3) If all result elements from step 1) checked in step 2) did not yield a ListenerDestination instance
2959 referencing the listener, then this use case is complete and the client knows that the listener is
2960 not presently represented by a listener destination within the implementation.

2961 8.7.3 Postconditions

2962 Unless errors occurred, the client either knows a ListenerDestination instance (including its object path)
2963 representing a listener destination within the implementation that references the particular listener, or
2964 knows that the listener is not referenced by any listener destination within the implementation.

2965 In the latter case, and if the implementation has also implemented the dynamic creation of listener
2966 destinations, the client could apply the CreateListenerDestination use case (see 8.8) to dynamically
2967 create a respective listener destination within the implementation that represents the desired listener.

2968 8.8 CreateListenerDestination: Create a new listener destination**2969 8.8.1 Preconditions**

2970 The client knows all of the following:

- 2971 • The same as for the SelectListenerDestination use case; see 8.7.1.

2972 8.8.2 Flow of activities

2973 1) Execute the SelectIndicationFilter use case (see 8.7).

2974 If a listener destination referencing the desired listener is found, use that; in this case, this use
2975 case is complete.

2976 2) Prepare a local instance of the CIM_ListenerDestination class that complies with the
2977 requirements of the ListenerDestination adaptation (see 7.3.23), inserting property values as
2978 follows:

2979 – Destination: the identification of the listener that the new listener destination is to
2980 reference, using the format required in 7.3.23.3.2. The format needs to be compatible
2981 with the requested protocol.

2982 – PersistenceType: the durability requested for the new listener destination, using the
2983 format required in 7.3.23.3.3.

2984 – Protocol: the protocol to used for the communication with the listener, using the format
2985 required by the CIM schema definition of the CIM_ListenerDestination class.

2986 3) Request the creation of the new listener destination in the implementation by invoking the
2987 CreateInstance() operation, providing the CIM_ListenerDestination instance prepared in step 2)
2988 as the actual value of the NewInstance parameter.

2989 If successful, the operation returns the object path of the ListenerDestination instance
2990 representing the newly created listener destination.

2991 If not successful, the operation returns a CIM status code providing details about the failure
2992 (see 7.3.23.3.4).

2993 8.8.3 Postconditions

2994 Unless errors occurred, the client knows the object path of a ListenerDestination instance representing a
2995 listener destination referencing the desired listener that either preexisted or was created; otherwise, the
2996 client knows details about why it was not possible to find or dynamically create the respective listener
2997 destination.

2998 8.9 FindFreeListenerDestination: Find a free listener destination

2999 8.9.1 Preconditions

3000 The client knows all of the following:

- 3001 • the object path to the IndicationService instance representing the indication service within the
3002 implementation (see 8.2)

3003 8.9.2 Flow of activities

3004 1) Execute the ListListenerDestinations use case (see 8.6).

3005 The result of this step is the set of ListenerDestination instances (including their object paths)
3006 representing all the listener destinations within the implementation.

3007 2) From the result of step 1), select a free listener destination; free listener destinations are
3008 represented by those ListenerDestination instances where the value of the Destination property
3009 is Null.

3010 8.9.3 Postconditions

3011 Unless errors occurred, the client knows a free listener destination, or knows that presently no free
3012 listener destinations exist within the implementation.

3013 8.10 ModifyListenerDestination: Modify an existing listener destination**3014 8.10.1 Preconditions**

3015 The client knows all of the following:

- 3016 • a local copy of a ListenerDestination instance (see 7.3.23)

3017 NOTE For example, the listener destination and its representing ListenerDestination instance might have been
3018 obtained by executing the FindFreeListenerDestination use case described in 8.9.

3019 8.10.2 Flow of activities

- 3020 1) Modify the local copy of the ListenerDestination instance, maintaining compliance with the
3021 requirements of the ListenerDestination adaptation (see 7.3.23).
- 3022 2) Modify the listener destination maintained by the implementation by invoking the
3023 ModifyInstance() operation, providing the CIM_ListenerDestination instance prepared in step 1)
3024 as the actual value of the ModifiedInstance parameter.

3025 If successful, the operation returns without error; otherwise, the operation returns a CIM status
3026 code providing details about the failure (see 7.3.23.3.6).

3027 8.10.3 Postconditions

3028 Unless errors occurred, the listener destination represented by the input ListenerDestination instance was
3029 modified; otherwise, the client knows details about why it was not possible to modify the represented
3030 listener destination.

3031 8.11 DeleteListenerDestination: Delete an existing listener destination**3032 8.11.1 Preconditions**

3033 The client knows all of the following:

- 3034 • the object path to a ListenerDestination instance (see 7.3.23)

3035 8.11.2 Flow of activities

- 3036 1) For the input ListenerDestination instance, find all AbstractSubscription instances (see 7.3.25)
3037 referencing the ListenerDestination instance (for example, using the ReferenceNames()
3038 operation).
- 3039 2) Delete all subscriptions referencing the input listener destination by executing the
3040 DeleteSubscription use case (see 8.21) for each AbstractSubscription instance returned by step
3041 1).
- 3042 3) Invoke the DeleteInstance() operation on the input ListenerDestination instance, effecting the
3043 deletion of the referenced listener destination.

3044 8.11.3 Postconditions

3045 Unless errors occurred, the input listener destination is deleted and no longer represented by any
3046 ListenerDestination instances.

3047 8.12 FindIndicationFilter: Find an indication filter covering a particular indication

3048 8.12.1 Preconditions

3049 The client knows all of the following:

- 3050 • the object path to the IndicationService instance representing the indication service within the
3051 implementation (see 7.3.2)
- 3052 • an implemented indication. Knowledge about whether or not a particular indication is actually
3053 implemented could for example be obtained by inspecting respective capabilities exposed by an
3054 implementation of a referencing profile that defines an adaptation of the particular indication.

3055 8.12.2 Flow of activities

3056 1) Find all indication filters within the responsibility of the indication service by traversing the
3057 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfIndicationFilter
3058 association adaptation (see 7.3.14) by invoking the Associators() operation with the following
3059 actual values for the input parameters:

- 3060 – InstanceName: the object path to the input IndicationService instance
- 3061 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
3062 IndicationServiceOfIndicationFilter association adaptation
- 3063 – ResultClass: "CIM_IndicationFilter", the adapted class of the IndicationFilter adaptation

3064 The result of this step is a set of IndicationFilter instances (see 7.3.11).

3065 2) Inspect each IndicationFilter instance resulting from step 1) by first checking the value of the
3066 QueryLanguage property. If the query language indicated by that value is interpretable by the
3067 client, interpret the query statement presented by the value of the Query property; otherwise,
3068 continue inspecting the next IndicationFilter instance returned by step 1).

3069 If the desired indication is not within the coverage as expressed by the query statement, then
3070 continue inspecting the next IndicationFilter instance returned by step 1).

3071 3) If the client desires to subscribe to the indication filter, continue by inspecting the IndicationFilter
3072 instance resulting from step 1) by checking whether the value of the
3073 IndividualSubscriptionSupported property is True. If so, this use case is complete; otherwise,
3074 continue with step 2) inspecting the next IndicationFilter instance returned by step 1); otherwise,
3075 this use case is complete.

3076 8.12.3 Postconditions

3077 Unless errors occurred, and if step 3) produced a suitable IndicationFilter instance, the client by that
3078 instance (including its object path) knows an indication filter that covers the desired indication and that
3079 supports individual subscriptions; otherwise, the client knows that within the responsibility of the indication
3080 service no such indication filter exists.

3081 8.13 DetermineQueryLanguages: Determine the set of query languages 3082 supported for query statements

3083 8.13.1 Preconditions

3084 The client knows all of the following:

- 3085 • The same as for the FindIndicationFilter use case described in 8.12.1.

3086 NOTE The procedure outlined in this use case is only an auxiliary approach to be pursued if preliminary
3087 knowledge about the query languages supported by an implementation is not available to the client.

3088 **8.13.2 Flow of activities**

- 3089 1) Execute steps 1) and 2) of the FindIndicationFilter use case (see 8.9), but vary step 2) to collect
3090 the query languages applied by all the inspected indication filters.

3091 **8.13.3 Postconditions**

3092 Unless errors occurred, the client knows all the query languages in use by existing indication filters.

3093 NOTE Because not all query languages supported by an implementation might be in use by indication filters, the
3094 set of query languages obtained by executing this use case is actually an open subset of the set of
3095 supported query languages.

3096 **8.14 CreateIndicationFilter: Create a dynamic indication filter covering a
3097 particular indication**3098 **8.14.1 Preconditions**

3099 The client knows all of the following:

- 3100 • The same as for the FindIndicationFilter use case described in 8.12.1.

3101 **8.14.2 Flow of activities**

- 3102 1) Execute the FindIndicationFilter use case (see 8.9).

3103 If a suitable indication filter covering the desired indication is found, use that; in this case, this
3104 use case is complete.

- 3105 2) If not already done previously, execute step 1) of the DetermineIndicationServiceCapabilities
3106 use case (see 8.4) and determine by the value of the FilterCreationEnabled property whether
3107 the support for dynamic indication filters as modeled by the DynamicIndicationFilters feature
3108 (see 7.2.1) is available.

- 3109 3) If the set of query languages supported by the implementation is not known a priori, execute the
3110 DetermineQueryLanguages use case (see 8.13).

- 3111 4) Prepare a local instance of the CIM_IndicationFilter class that complies with the requirements of
3112 the DynamicIndicationFilter adaptation (see 7.3.13), inserting property values as follows:

- 3113 – QueryLanguage: a query language supported by the implementation; see 7.3.11.3.6.
- 3114 – Query: the query statement covering the desired set of indications; see 7.3.11.3.5.

3115 NOTE Additional constraints on properties of the CIM_Indication class selected by the
3116 query statement may be specified through the WHERE clause; however, if the
3117 implementation is unable to comply with these constraints, the operation will fail.

- 3118 – SourceNamespaces[]: a list of local namespace names identifying the namespaces
3119 considered as ; see 7.3.11.3.3.

- 3120 5) Request the creation of the new dynamic indication filter in the implementation by invoking the
3121 CreateInstance() operation, providing the CIM_IndicationFilter instance prepared in step 4) as
3122 the actual value of the NewInstance parameter.

3123 If successful, the operation returns the object path of the DynamicIndicationFilter instance
3124 representing the newly created dynamic indication filter.

3125 If not successful, the operation returns a CIM status code providing details about the failure
3126 (see 7.3.13.2.2).

3127 8.14.3 Postconditions

3128 Unless errors occurred, the client knows the object path of an IndicationFilter instance representing an
3129 indication filter covering the desired indication that either preexisted or was dynamically created;
3130 otherwise, the client knows details about why it was not possible to find or dynamically create the
3131 respective indication filter.

3132 8.15 ModifyIndicationFilter: Modify a dynamic indication filter

3133 8.15.1 Preconditions

3134 The client knows all of the following:

- 3135 • a local copy of an DynamicIndicationFilter instance (see 7.3.13)

3136 NOTE For example, that dynamic indication filter and its representing DynamicIndicationFilter instance might
3137 have been created by executing the CreateIndicationFilter use case; see 8.14.

3138 8.15.2 Flow of activities

- 3139 1) Modify the local copy of the DynamicIndicationFilter instance, maintaining compliance with the
3140 requirements of the DynamicIndicationFilter adaptation (see 7.3.13).
- 3141 2) Modify the dynamic indication filter maintained by the implementation by invoking the
3142 ModifyInstance() operation, providing the DynamicIndicationFilter instance prepared in step 1)
3143 as the actual value of the ModifiedInstance parameter.
- 3144 3) If successful, the operation returns without error; otherwise, the operation returns a CIM status
3145 code providing details about the failure (see 7.3.13.2.4).

3146 8.15.3 Postconditions

3147 Unless errors occurred, the dynamic indication filter represented by the input DynamicIndicationFilter
3148 instance was modified; otherwise, the client knows details about why it was not possible to modify the
3149 represented dynamic indication filter.

3150 8.16 DeleteIndicationFilter: Delete a dynamic indication filter

3151 8.16.1 Preconditions

3152 The client knows all of the following:

- 3153 • the object path to a DynamicIndicationFilter instance (see 7.3.13)

3154 8.16.2 Flow of activities

- 3155 1) For the input DynamicIndicationFilter instance, find all AbstractSubscription instances (see
3156 7.3.25) referencing the DynamicIndicationFilter instance (for example, using the
3157 ReferenceNames() operation).
- 3158 2) Delete all subscriptions referencing the input listener destination, by executing the
3159 DeleteSubscription use case (see 8.21) for each AbstractSubscription instance returned by step
3160 1).
- 3161 3) Invoke the DeleteInstance() operation on the input DynamicIndicationFilter instance, effecting
3162 the deletion of the referenced dynamic indication filter.

3163 8.16.3 Postconditions

3164 Unless errors occurred, the input dynamic indication filter is deleted and no longer represented by any
3165 DynamicIndicationFilter instances.

3166 8.17 CheckCollectionCoverage: Check the coverage of a filter collection**3167 8.17.1 Preconditions**

3168 The client knows all of the following:

- 3169 • a local copy of a StaticFilterCollection instance (see 7.3.17), and the object path referencing the
3170 original StaticFilterCollection instance within the implementation

3171 8.17.2 Flow of activities

3172 1) Check whether the input filter collection contains any elements by resolving — from the
3173 StaticFilterCollection instance — the CIM_ConcreteComponent association as modeled by the
3174 IndicationFilterInFilterCollection association adaptation (see 7.3.19) and the
3175 FilterCollectionInFilterCollection association adaptation (see 7.3.20).

3176 If no contained elements are discovered, a defined coverage may apply as the coverage; in this
3177 case, skip to step 4).

3178 2) For each of the contained elements found in step 1), determine the contributed coverage and
3179 add that to the resulting aggregated coverage of the input filter collection.

3180 In the case of a contained indication filter, the contributed coverage is determined by inspecting
3181 the values of the QueryLanguage property and that of the Query property containing the query
3182 statement.

3183 In the case of a contained filter collection, the contributed coverage is determined by recursively
3184 applying this use case (8.17).

3185 3) Aggregate the contributed coverage of each contained element as determined in step 2) into the
3186 resulting aggregated coverage of the input filter collection. After completing this step the client
3187 knows the aggregated coverage of the input filter collection, and this use case is complete.

3188 4) This step applies if no contained elements were discovered in steps 2) and 3).

3189 Check the value of the CollectionName property in the StaticFilterCollection instance for the
3190 pattern required for the name the global filter collection covering all instance lifecycle
3191 indications, as detailed in 7.3.22.4.4.

3192 If the pattern matches, the client knows that the represented filter collection is the global filter
3193 collection covering all instance lifecycle indications; in this case, the client knows that the
3194 coverage of the input filter collection is all instance lifecycle indications and this use case is
3195 complete.

3196 5) Check the value of the CollectionName property in the StaticFilterCollection instance for the
3197 pattern required for the name of global filter collections for profile defined indications, as defined
3198 in 7.3.22.

3199 If the pattern matches, the client knows that the represented filter collection is a global filter
3200 collection for profile defined indications with a defined coverage as detailed in 7.3.22. The client
3201 needs to have a priori knowledge about the defined coverage of each referencing profile, and
3202 this use case is complete.

3203 6) Check the value of the CollectionName property in the StaticFilterCollection instance for the
3204 pattern required for the name of profile-specific filter collections as defined in 7.3.21.2.2.

3205 If the pattern matches, the client knows that the input filter collection is a profile-specific filter
 3206 collection with a defined coverage as detailed in 7.3.21.3. The client needs to have a priori
 3207 knowledge about the defined coverage of the identified referencing profile, and this use case is
 3208 complete.

3209 7) If the input filter collection does not match any of the types determined in steps 4), 5), and 6),
 3210 then no defined coverage applies. Furthermore, because no contained elements were
 3211 discovered in step 2), the coverage of the input filter collection is empty (that is, it does not
 3212 cover any indications).

3213 8.17.3 Postconditions

3214 Unless errors occurred, or in the cases determined in steps 5) and 6) above the client does not have a
 3215 priori knowledge about the defined coverage(s), the client knows the coverage of the input filter collection.

3216 8.18 ObtainNamedCollection: Obtain a named filter collection

3217 8.18.1 Preconditions

3218 The client knows all of the following:

- 3219 • the object path to the IndicationService instance representing the indication service within the
 3220 implementation (see 7.3.2)
- 3221 • the name of the named filter collection, for example, the name of a global filter collection or of a
 3222 profile-specific filter collection

3223 8.18.2 Flow of activities

3224 1) Find all filter collections within the responsibility of the indication service by traversing the
 3225 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfFilterCollection
 3226 association adaptation (see 7.3.18) by invoking the Associators() operation with the following
 3227 actual values for the input parameters:

- 3228 – InstanceName: the object path to the input IndicationService instance
- 3229 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
 3230 IndicationServiceOfFilterCollection association adaptation
- 3231 – ResultClass: "CIM_FilterCollection", the adapted class of the StaticFilterCollection
 3232 adaptation

3233 The result of this step is a set of StaticFilterCollection instances (see 7.3.17).

3234 2) Inspect each StaticFilterCollection instance resulting from step 1) by checking the value of the
 3235 CollectionName property. If the name of the static filter collection as indicated by that value
 3236 matches the desired name, this use case is complete; otherwise, continue inspecting the next
 3237 IndicationFilter instance returned by step 1).

3238 8.18.3 Postconditions

3239 Unless errors occurred, the client knows the named filter collection by means of the representing
 3240 StaticFilterCollection instance (including its object path).

3241 8.19 CreateSubscription: Create a subscription

3242 8.19.1 Preconditions

3243 The client knows all of the following:

3244 • the object path to the IndicationService instance representing the indication service within the
3245 implementation (see 7.3.2)

3246 • an object path to an IndicationFilter instance representing an indication filter covering the
3247 desired indication or set of indications

3248 For example, see the FindIndicationFilter (8.12) or CreateIndicationFilter (8.14) use cases about
3249 how to obtain that object path.

3250 • Alternatively, an object path to a StaticFilterCollection instance representing a filter collection
3251 covering the desired indication or set of indications. For example, see the
3252 ObtainNamedCollection use case (8.18) about how to obtain the object path to a
3253 StaticFilterCollection instance representing a global filter collection or a profile-specific filter
3254 collection.

3255 • an object path to a ListenerDestination instance representing a listener destination that
3256 represents the desired listener within the implementation. For example, see the
3257 SelectListenerDestination use case (8.7) about how to obtain that object path.

3258 8.19.2 Flow of activities

3259 1) Prepare a local instance of the CIM_IndicationSubscription class (or the
3260 CIM_FilterCollectionSubscription for a subscription to a filter collection) that complies with the
3261 requirements of the FilterSubscription adaptation (see 7.3.26) or the CollectionSubscription
3262 adaptation (see 7.3.27), inserting property values as follows:

3263 – Filter: input object path to the indication filter (or to the filter collection)

3264 – Handler: input object path to the listener destination

3265 The values of other properties should be specified in conformance with the capabilities of the
3266 implementation as exposed by instances of the IndicationService adaptation and the
3267 IndicationServiceCapabilities adaptation; see the DetermineIndicationServiceCapabilities use
3268 case (8.4) to obtain knowledge about these capabilities.

3269 Values not described through these adaptations may or may not be respected by the
3270 implementation; in this case it is implementation dependent whether in step 2) the
3271 implementation imposes a respective default behavior, or whether it fails in creating the new
3272 subscription.

3273 2) Define the new subscription to the implementation by invoking the CreateInstance() operation,
3274 providing the CIM_IndicationSubscription (or CIM_FilterCollectionSubscription) instance
3275 prepared in step 1) as the actual value of the NewInstance parameter.

3276 If successful, the operation returns the object path of the DynamicIndicationFilter instance
3277 representing the newly created subscription.

3278 If not successful, the operation returns a CIM status code providing details about the failure
3279 (see 7.3.26.3.2 or 7.3.27.3.2).

3280 **8.19.3 Postconditions**

3281 Unless errors occurred, the client knows the object path of an AbstractSubscription instance representing
 3282 the newly created subscription; otherwise, the client knows details about why it was not possible to create
 3283 the subscription.

3284 **8.20 CheckSubscriptions: Determine whether subscriptions exist for a given**
 3285 **indication and listener**3286 **8.20.1 Preconditions**

3287 The client knows all of the following:

- 3288 • the object path to the IndicationService instance representing the indication service within the
 3289 implementation (see 8.2)
- 3290 • the URI exposed by the desired listener

3291 **8.20.2 Flow of activities**

- 3292 1) Execute the ListListenerDestinations use case (see 8.6). The result is a set of
 3293 ListenerDestination instances (including their object paths) representing all the listener
 3294 destinations within the implementation.
 - 3295 2) From the result of step 1), drop all ListenerDestination instances not referencing the desired
 3296 listener. The result is a set of ListenerDestination instances (including their object paths)
 3297 representing all the listener destinations referencing the desired listener.
 - 3298 3) For each ListenerDestination instance resulting from step 2), find all IndicationFilter instances
 3299 (see 7.3.11) associated with the ListenerDestination instance (see 7.3.23) through a
 3300 FilterSubscription instance (see 7.3.26). The result of this step is a set of IndicationFilter
 3301 instances representing indication filters to which the desired listener is subscribed.
 - 3302 4) Inspect each IndicationFilter instance resulting from step 3) by checking the values of the
 3303 QueryLanguage and the Query properties. Interpret the query statement expressed by the value
 3304 of the Query property and check whether the input indication is covered. If the input indication is
 3305 covered, add the identification of the represented listener destination to a filter result list, and
 3306 continue inspecting the next IndicationFilter instance returned by step 3).
 - 3307 5) For each ListenerDestination instance resulting from step 2), find all StaticFilterCollection
 3308 instances (see 7.3.17) associated through a CollectionSubscription instance (see 7.3.27). The
 3309 result of this step is a set of StaticFilterCollection instances representing static filter collections
 3310 to which the desired listener is subscribed.
 - 3311 6) For each StaticFilterCollection instance resulting from step 5), apply the
 3312 CheckCollectionCoverage use case (see 8.17).
- 3313 If the input indication is covered, add the identification of the represented static filter collection to
 3314 a collection result list, and continue inspecting the next StaticFilterCollection instance returned
 3315 by step 5).

3316 **8.20.3 Postconditions**

3317 Unless errors occurred, the client knows (the identifications of) all listener destinations and filter
 3318 collections to which the desired listener is subscribed.

3319 8.21 DeleteSubscription: Delete a subscription

3320 8.21.1 Preconditions

3321 The client knows all of the following:

- 3322 • the object path to the AbstractSubscription instance (see 7.3.25) representing a subscription
- 3323 within the implementation

3324 8.21.2 Flow of activities

- 3325 1) Invoke the DeleteInstance() operation on the AbstractSubscription instance, effecting the
- 3326 deletion of the represented subscription.

3327 NOTE If the subscription referenced a dynamic indication filter, and no other subscriptions reference it, and the
3328 client does not plan to create a new subscription for this filter, the client can delete the dynamic indication
3329 filter using the DeleteFilter use case (see 8.16); likewise, unless referenced by other subscriptions, the
3330 client can delete the listener destination that was referenced by the deleted subscription, using the
3331 DeleteListenerDestination use case (see 8.11).

3332 8.21.3 Postconditions

3333 Unless errors occurred, the subscription is deleted and no longer represented by any
3334 AbstractSubscription instance.

3335 8.22 FindAlertingSystem: Find the system containing a component causing an 3336 alert indication

3337 8.22.1 Preconditions

3338 The client knows all of the following:

- 3339 • an AlertIndication instance representing an alert indication that references the alerting managed
- 3340 element

3341 8.22.2 Flow of activities

- 3342 1) Obtain the CIM element referenced by the value of the AlertingManagedElement in the input
- 3343 AlertIndication instance.
- 3344 2) Determine the profile with which the CIM element is conformant and where the central class
- 3345 adaption adapts the CIM_System class.

3346 NOTE This step implies client knowledge about profiles defining adaptations of the class of the CIM
3347 element obtained in step 1). More than one profile could impact the CIM element, but the
3348 scoping CIM_System instance should be the same in all cases.

- 3349 3) Use the scoping algorithm defined by the profile determined in step 2) to find the related
- 3350 instance of the scoping class adaptation of that profile.

3351 8.22.3 Postconditions

3352 Unless errors occurred, the client knows the CIM_System instance representing the system containing a
3353 component causing the generation of the input alert indication.

3354 8.23 DetermineIndicationGate: Determine the indication gate of an indication

3355 8.23.1 Preconditions

3356 The client knows all of the following:

- 3357 • an AlertIndication instance representing an alert indication that references the alerting managed
3358 element

3359 In addition, subscriptions for the listener that received the input alert indication should have been
3360 established such that within the set of subscribed to indication gates within a particular implementation
3361 each is uniquely identified with a name as exposed by the value of the Name property in representing
3362 IndicationFilter instances (see 7.3.11), or as exposed by the value of the CollectionName property in
3363 representing StaticFilterCollection instances (see 7.3.17).

3364 NOTE This policy ensures that indication gate names are unique with respect to one implementation;
3365 implementations are unable to (and not required to) maintain that uniqueness, but clients can ensure it
3366 through carefully applying the subscription policy stated above for each listener that a client controls.

3367 8.23.2 Flow of activities

- 3368 1) Extract the value of the IndicationFilterName from the input AlertIndication instance as the name
3369 of the sought-after indication gate.

3370 If the input alert indication originates from an implementation that is known to the client by
3371 reference to its representing IndicationFilter instance, skip to step 8); otherwise, continue with
3372 step 2).

- 3373 2) Inspect the value of the AlertingManagedElement property of the input AlertIndication instance.

3374 If that value is Null, then the indication gate cannot be determined, and this use case is
3375 complete without success; this is also the case of the value is a URI that does not reference a
3376 CIM instance that represents the alerting managed element. In subsequent steps it is assumed
3377 that the value is a URI that references a CIM instance that represents the alerting managed
3378 element.

- 3379 3) Determine the ProfileRegistration instance that is providing the CIM instance referenced by the
3380 URI found in step 2), using one of the algorithms described in [DSP1033](#) for that purpose.

- 3381 4) Apply the LocateProfileIndicationService use case (see 8.3) in order to determine the
3382 IndicationService instance (see 7.3.2) that represents the indication service from which the input
3383 alert indication originated.

- 3384 5) Find all IndicationFilter instances (see 7.3.11) associated with the IndicationFilter instance (see
3385 7.3.23) found in step 4) through an IndicationServiceOfIndicationFilter instance (see 7.3.14), for
3386 example by executing the Associators() operation.

- 3387 6) For each IndicationFilter instance obtained in step 5), determine if the value of the Name
3388 property matches the name of the sought-after indication gate determined in step 1).

3389 If it matches, and the subscription policy mentioned in the preconditions was maintained, then
3390 the indication filter represented by the IndicationFilter instance is the sought-after indication
3391 gate.

3392 If the name matches, and the subscription policy was not maintained, then all IndicationFilter
3393 instances determined in step 5) need to be checked with step 6) in order to ensure that the
3394 name as exposed by the value of the Name property is not used more than once. If this is the
3395 case, the sought-after indication gate cannot be exactly determined; however, at least it can be
3396 limited to the set of indication filters using the name as determined in step 1).

- 3397 If a name does match, continue with step 8).
- 3398 If the name does not match, the next instance from the set determined in step 5) needs to be
3399 checked with step 6); if no additional instances remain, continue with step 7).
- 3400 7) Repeat steps 5) and 6) for filter collections, searching for StaticFilterCollection instances (see
3401 7.3.17) associated through an IndicationServiceOfFilterCollection instance (see 7.3.18) in step
3402 5), and checking the value of the CollectionName property in step 6).
- 3403 8) If an indication filter was determined as the sought-after indication gate in steps 1), 6), or 7), the
3404 client can check the query statement exposed by the value of the Query property in the
3405 representing IndicationFilter instance (or — in case the alert indication was received through a
3406 filter collection — in at least one of the contained IndicationFilter instances), and verify that the
3407 input alert indication is indeed within the coverage of the identified indication filter or filter
3408 collection.

3409 8.23.3 Postconditions

3410 Unless errors occurred, the client knows the indication gate emitting the input alert indication by means of
3411 its representing IndicationFilter or StaticFilterCollection instance.

3412 8.24 SubscribeForProfileIndications: Subscribe for all of the indications defined 3413 in a referencing profile

3414 8.24.1 Preconditions

3415 The client knows the following:

- 3416 • the registered name of the referencing profile
- 3417 • the object path to the IndicationService instance representing the indication service within the
3418 implementation (see 7.3.2)
- 3419 • the object path to the ListenerDestination instance (see 7.3.23) representing the desired listener
3420 destination

3421 8.24.2 Flow of activities

- 3422 1) Construct the name for the profile-specific filter collection for alert indications, applying the
3423 pattern defined in 7.3.21.2.2.
- 3424 2) Execute the ObtainNamedCollection use case (see 8.18), providing the name constructed in
3425 step 1) as input; the result is either Null or the object path referencing the
3426 ProfileSpecificAlertIndicationFilterCollection instance (see 7.3.21) representing the profile-
3427 specific filter collection for alert indications of the referencing profile.
- 3428 3) If an object path was returned on step 2), execute the CreateSubscription use case (see 8.19),
3429 providing that object path and the input object path to the ListenerDestination instance as input.
- 3430 4) Perform steps 1), 2) and 3) analogously for lifecycle indications.

3431 8.24.3 Postconditions

3432 Unless errors occurred, the desired listener destination is subscribed for all alert indications and all
3433 lifecycle indications defined by the referencing profile.

3434

ANNEX A (informative)

3435
3436
3437
3438

Profiles defining indications

3439 Referencing profiles define indications and related requirements in the following ways:

- 3440 • Reference this profile as a mandatory or conditional profile
- 3441 • Define lifecycle indications and/or alert indications by defining adaptations based on the
3442 LifecycleIndication adaptation (see 7.3.32) and/or the AlertIndication adaptation (see 7.3.31).
3443 This requires but is not limited to defining the requirement level, the reported event, and the
3444 query statement; however, the latter two may be implied by the respective base adaptation.
- 3445 • Optionally, define indication filters by defining adaptations based on the StaticIndicationFilter
3446 adaptation (see 7.3.11). The definition of indication-specific indication filters covering each
3447 lifecycle indication and each alert indication defined in a referencing profile is implied by this
3448 profile through the IndicationSpecificIndicationFilter adaptation (see 7.3.15), but may be refined
3449 by referencing profiles.
- 3450 • Optionally, define filter collections by defining adaptations based on the StaticFilterCollection
3451 adaptation (see 7.3.17). The definition of profile-specific filter collections covering all lifecycle
3452 indications and/or alert indications defined in a referencing profile is implied by this profile
3453 through the ProfileSpecificFilterCollection adaptation (see 7.3.21), but may be refined by
3454 referencing profiles.

ANNEX B (informative)

Change log

3455
3456
3457
3458

Version	Date	Description
1.0.0	2008-12-05	
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)
1.1.0	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.
1.2.0	2011-06-30	Released as a DMTF Standard, with the following changes: <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	Released as a DMTF Standard, with the following errata corrected: <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.
1.2.2	2014-04-24	Released as DMTF Standard with the following errata corrected: <ul style="list-style-type: none"> • Fixed use of incorrect status code CIM_ERR_NOT_IMPLEMENTED to CIM_ERR_NOT_SUPPORTED • Changed the requirement for GlobalIndicationFilter for lifecycle indications to an optional feature: LifeCycleGlobalIndicationFilter (see 7.2.9) • Changed the requirement for GlobalIndicationFilter for alert indications to an optional feature: AlertGlobalIndicationFilter (see 7.2.10) • Updated the operation names as per DSP0223 1.0.2 • Fixed editorial issues

3459