



1
2
3
4
5

Document Number: DSP1054

Date: 2011-06-30

Version: 1.2.0

6 **Indications Profile**

7 **Document Type: Specification**

8 **Document Status: DMTF Standard**

9 **Document Language: en-US**

10

11 Copyright notice

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

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

CONTENTS

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

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

141	8.10	ModifyListenerDestination: Modify an existing listener destination	103
142	8.11	DeleteListenerDestination: Delete an existing listener destination.....	103
143	8.12	FindIndicationFilter: Find an indication filter covering a particular indication	104
144	8.13	DetermineQueryLanguages: Determine the set of query languages supported for query	
145		statements	104
146	8.14	CreateIndicationFilter: Create a dynamic indication filter covering a particular indication	105
147	8.15	ModifyIndicationFilter: Modify a dynamic indication filter.....	106
148	8.16	DeleteIndicationFilter: Delete a dynamic indication filter	106
149	8.17	CheckCollectionCoverage: Check the coverage of a filter collection	107
150	8.18	ObtainNamedCollection: Obtain a named filter collection	108
151	8.19	CreateSubscription: Create a subscription	108
152	8.20	CheckSubscriptions: Determine whether subscriptions exist for a given indication and	
153		listener	110
154	8.21	DeleteSubscription: Delete a subscription	110
155	8.22	FindAlertingSystem: Find the system containing a component causing an alert indication...	111
156	8.23	DetermineIndicationGate: Determine the indication gate of an indication.....	111
157	8.24	SubscribeForProfileIndications: Subscribe for all of the indications defined in a	
158		referencing profile	113
159	ANNEX A (informative)	Profiles defining indications	114
160	ANNEX B (informative)	Change Log.....	115
161			
162	Figures		
163	Figure 1 –	Indication related functionality within an implementation	22
164	Figure 2 –	Indications Profile: DMTF class adaptation diagram	31
165	Figure 3 –	Indications Profile: Indication adaptations and adapted indication classes.....	33
166	Figure 4 –	DMTF object diagram: Global and profile-specific filter collections.....	93
167	Figure 5 –	DMTF object diagram: Filter collections and contained indication filters.....	95
168	Figure 6 –	DMTF object diagram: Static listener destinations	96
169			
170	Tables		
171	Table 1 –	Profile references	17
172	Table 2 –	Adaptations	17
173	Table 3 –	Features	18
174	Table 4 –	IndicationService: Element requirements	38
175	Table 5 –	ModifyInstance(): Error reporting requirements	39
176	Table 6 –	IndicationSystem: Element requirements	41
177	Table 7 –	HostedIndicationService: Element requirements.....	41
178	Table 8 –	IndicationsProfileRegistration: Element requirements	42
179	Table 9 –	ElementConformsToProfile: Element requirements	43
180	Table 10 –	IndicationServiceCapabilities: Element requirements.....	44
181	Table 11 –	CapabilitiesOfIndicationService: Element requirements.....	45
182	Table 12 –	IndicationServiceInitialSettings: Element requirements.....	46
183	Table 13 –	InitialSettingsOfIndicationService: Element requirements.....	47
184	Table 14 –	IndicationFilter: Element requirements	48
185	Table 15 –	StaticIndicationFilter: Element requirements	50
186	Table 16 –	DynamicIndicationFilter: Element requirements	51
187	Table 17 –	CreateInstance(): Error reporting requirements.....	51
188	Table 18 –	DeleteInstance(): Error reporting requirements	53

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

190 Table 20 – IndicationServiceOfIndicationFilter: Element requirements 54

191 Table 21 – IndicationSpecificIndicationFilter: Element requirements 55

192 Table 22 – GlobalIndicationFilter: Element requirements 57

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

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

195 Table 25 – StaticFilterCollection: Element requirements 59

196 Table 26 – IndicationServiceOfFilterCollection: Element requirements 60

197 Table 27 – IndicationFilterInFilterCollection: Element requirements 61

198 Table 28 – FilterCollectionInFilterCollection: Element requirements 62

199 Table 29 – ProfileSpecificFilterCollection: Element requirements 62

200 Table 30 – GlobalFilterCollection: Element requirements 64

201 Table 31 – ListenerDestination Element requirements 67

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

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

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

205 Table 35 – IndicationServiceOfListenerDestination: Element requirements 70

206 Table 36 – AbstractSubscription: Element requirements 71

207 Table 37 – RepeatNotificationPolicy: Value constraints 72

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

209 Table 39 – FilterSubscription: Element requirements 75

210 Table 40 – CreateInstance(): Error reporting requirements 75

211 Table 41 – CollectionSubscription: Element requirements 77

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

213 Table 43 – ProfileOfFilterCollection: Element requirements 78

214 Table 44 – BasicIndication: Element requirements 80

215 Table 45 – ReliableIndication: Element requirements 81

216 Table 46 – AlertIndication: Element requirements 84

217 Table 47 – LifecycleIndication: Element requirements 87

218 Table 48 – ListenerDestinationRemovalIndication: Element requirements 88

219 Table 49 – SubscriptionRemovalIndication: Element requirements 89

220

221

Foreword

222 The *Indications Profile* (DSP1054) was prepared by the DMTF WBEM Infrastructure Modeling Working
223 Group. Version 1.0 was prepared by the DMTF WBEM Infrastructure and Protocols Working Group.

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

226 Acknowledgments

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

228 Editor:

- 229 • Michael Johanssen (IBM)

230 Contributors:

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

240

Introduction

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

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

247 Document conventions

248 Typographical conventions

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

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

255 ABNF usage conventions

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

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

260 Deprecated material

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

266 The following typographical convention indicates deprecated material:

267 DEPRECATED

268 Deprecated material appears here.

269 DEPRECATED

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

272 Experimental material

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

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

278 The following typographical convention indicates experimental material:

279 **EXPERIMENTAL**

280 Experimental material appears here.

281 **EXPERIMENTAL**

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

284

Indications Profile

285 1 Scope

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

289 2 Normative references

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

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

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

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

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

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

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

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

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

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

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

314 **3 Terms and definitions**

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

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

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

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

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

331 **3.1**

332 **alert indication**

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

335 **3.2**

336 **client**

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

339 **3.3**

340 **coverage**

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

343 **3.4**

344 **defined coverage**

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

347 **3.5**

348 **dynamic indication filter**

349 an indication filter whose lifecycle is controlled by a client

350 **3.6**

351 **event**

352 an observable occurrence of a phenomenon of interest
353 For details, see 6.1.

- 354 **3.7**
355 **filter collection**
356 an indication gate that may contain other indication gates such as indication filters or other filter
357 collections
358 For details, see 6.3.
- 359 **3.8**
360 **global indication filter**
361 an indication filter that covers large sets of indications, such as all alert indications
362 For details, see 6.2.5.
- 363 **3.9**
364 **global filter collection**
365 a filter collection that covers large sets of indications, such as all lifecycle indications
366 For details, see 6.3.3.5.
- 367 **3.10**
368 **implementation**
369 a WBEM server that implements applicable portions of this profile and of referencing profiles
370 For details, see [DSP1001](#).
- 371 **3.11**
372 **indication**
373 the notification about an event that occurred
374 For details, see 6.1.
- 375 **3.12**
376 **indication delivery**
377 the process of delivering indications from an implementation to a listener
378
- 379 **indication filter**
380 an indication gate whose coverage is defined through a query statement
381 For details, see 6.2
- 382 **3.13**
383 **indication filtering**
384 the process of selecting indications based on filtering rules applied by indication gates, such that only
385 indications within the coverage of the indication gate pass the indication gate
- 386 **3.14**
387 **indication gate**
388 a managed element that filters indications such that only indications within its coverage pass. Indication
389 gates can serve as targets for subscriptions, and control which indications are delivered to subscribed
390 listeners.
- 391 **3.15**
392 **indication generation**
393 the process of creating an indication as the event that the indication is designed to report occurs
- 394 **3.16**
395 **indication origin**
396 the namespace out of that the indication originates
397 For details, see 6.1.2.4.

- 398 **3.17**
399 **indication service**
400 a component within a WBEM server for indication related processing, including handling of subscriptions
401 and delivery of indications to a WBEM listener
- 402 **3.18**
403 **indication system**
404 a system that hosts a WBEM server with one or more indication services
405 For details, see 6.6.
- 406 **3.19**
407 **indication-specific indication filter**
408 a static indication filter that covers a particular indication specified in a profile
409 For details, see 6.2.4.
- 410 **3.20**
411 **Interop namespace**
412 a namespace containing CIM instances representing specific capabilities of a WBEM server
413 Examples include CIM_RegisteredProfile instances representing specific versions of profiles or
414 CIM_IndicationFilter instances representing indication filters. For details, see [DSP1033](#).
- 415 **3.21**
416 **lifecycle indication**
417 an indication indicating an event related to the lifecycle of CIM instances or CIM classes; for details,
418 see 6.1.2.3.
- 419 **3.22**
420 **listener**
421 a WBEM listener that implements applicable portions of this profile
422 For details, see [DSP1001](#).
- 423 **3.23**
424 **listener destination**
425 an entity that maintains a reference to a listener within an implementation; for details, see 6.4.5..
- 426 **3.24**
427 **profile-specific filter collection**
428 a static filter collection that covers all indications of a particular type defined in a profile
429 For details, see 6.3.3.4.
- 430 **3.25**
431 **query statement**
432 a statement expressed in a query language used to describe either (a part of) an event or the coverage of
433 an indication filter
- 434 **3.26**
435 **referencing profile**
436 a profile referencing this profile
437 Note that [DSP1001](#) requires each profile that defines indications to reference this profile.
- 438 **3.27**
439 **reliable indication**
440 an indication containing a sequence identifier enabling listeners to detect duplicate, missing, or out-of-
441 order indications

- 442 For details, see 6.1.5 and 7.4.
- 443 **3.28**
444 **repeated indication**
445 an indication that reports the same event as a previous indication
446 For details, see 6.1.6.
- 447 **3.29**
448 **repeated indication delivery**
449 the delivery of repeated indications
450 Repeated indication delivery typically occurs if the reported event describes a persistent situation such as
451 exceeding a threshold value.
- 452 **3.30**
453 **sequence identifier**
454 data element with a reliable indication that ensures unique identification of the reliable indication
455 A sequence identifier is composed of a sequence context and a sequence number
456 For details, see 7.4.2.
- 457 **3.31**
458 **sequence identifier lifetime**
459 a maximum time interval maintained by an implementation implementing reliable indications within which
460 the implementation retries failed indication delivery attempts
461 For details, see 7.4.2.
- 462 **3.32**
463 **static filter collection**
464 a filter collection whose lifecycle is controlled by the implementation, that is uniquely identifiable and for
465 which a defined coverage is established
466 For details, see 6.3.3.
- 467 **3.33**
468 **static indication filter**
469 an indication filter whose lifecycle is controlled by the implementation
- 470 **3.34**
471 **subscription**
472 the mechanism whereby a client registers a listener for the delivery of indications from an implementation
- 473 **3.35**
474 **this profile**
475 a short term for the Indications profile, the profile specified in this specification document (DSP1054)
- 476 **3.36**
477 **WBEM client**
478 a CIM client (see [DSP0004](#)) that supports a WBEM protocol
479 For details, see [DSP1001](#).
- 480 **3.37**
481 **WBEM listener**
482 a CIM listener (see [DSP0004](#)) that supports a WBEM protocol
483 For details, see [DSP1001](#).

484 **3.38**
485 **WBEM server**
486 a CIM server (see [DSP0004](#)) that supports a WBEM protocol
487 For details, see [DSP1001](#).

488 **4 Symbols and abbreviated terms**

489 **4.1**
490 **CQL**
491 CIM Query Language

492 **4.2**
493 **QoS**
494 Quality of service

495 **4.3**
496 **URI**
497 Uniform Resource Identifier

498 **4.4**
499 **WBEM**
500 Web Based Enterprise Management

501 **5 Synopsis**

502 **Profile name:** Indications

503 **Version:** 1.2.0

504 **Organization:** DMTF

505 **Profile type:** Component

506 **Schema version:** 2.25

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

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

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

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

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

515 **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.

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

517 **Table 2 – Adaptations**

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

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

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

519 **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.

520 6 Description

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

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

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

532 6.1 Events and indications

533 6.1.1 Events

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

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

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

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

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

543 6.1.2 Indications

544 6.1.2.1 General

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

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

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

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

557 6.1.2.2 Alert indications

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

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

562 6.1.2.3 Lifecycle indications

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

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

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

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

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

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

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

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

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

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

613 **6.1.2.4 Origin of indications**

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

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

619 **6.1.3 Definition of events and indications in referencing profiles**

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

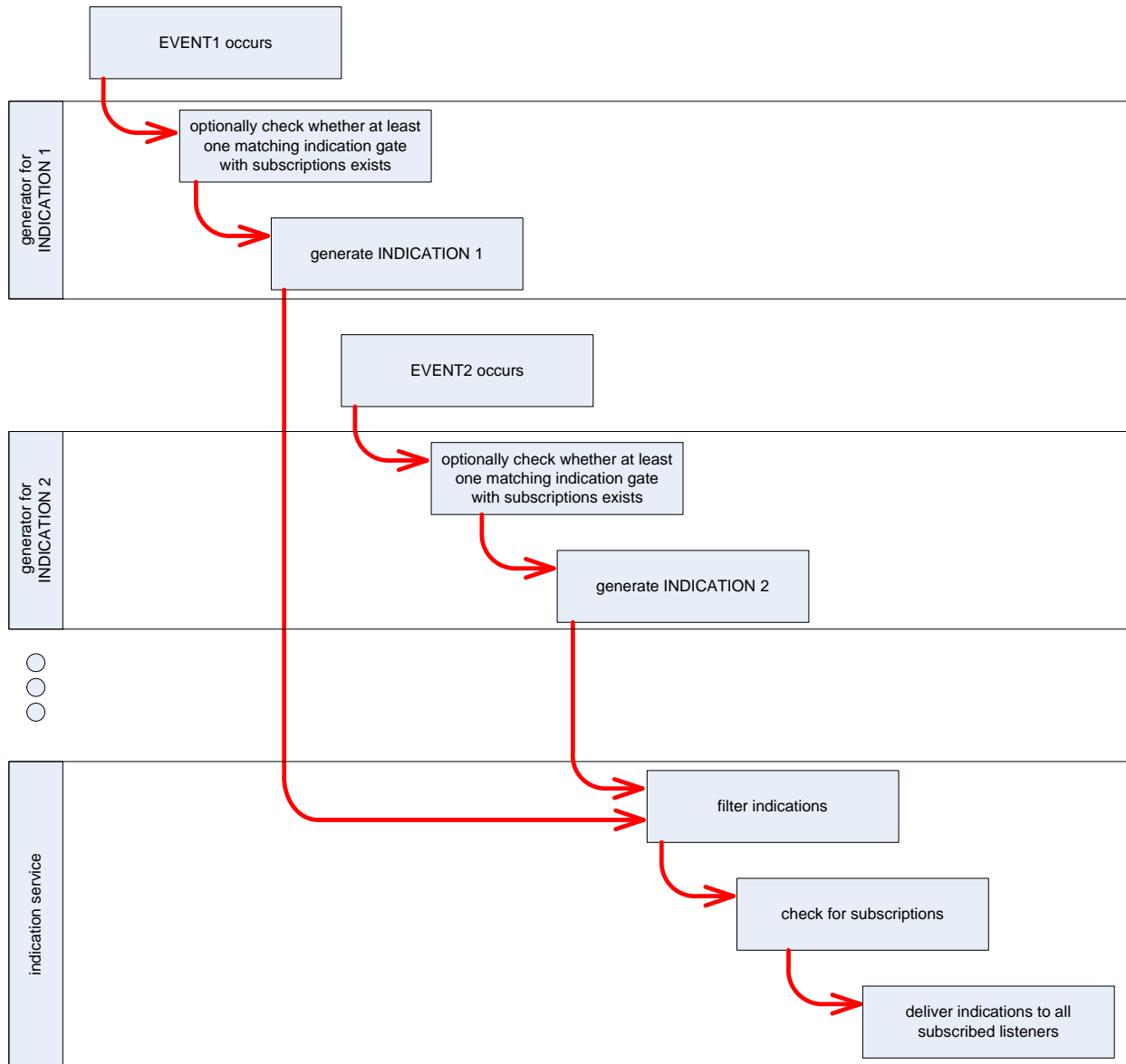
622 NOTE Defining events separately is particularly useful if multiple indications reporting the same event are
623 defined. However, if an event is only reported through one indication, the event definition as part of the
624 definition of the indication adaptation is more compact.

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

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

636 **6.1.4 Indication generation, indication filtering, and indication delivery**

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



639

640

Figure 1 – Indication related functionality within an implementation

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

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

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

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

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

660 **6.1.5 Reliable indication delivery**

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

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

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

667 **6.1.6 Avoidance of repeated indication delivery**

668 **6.1.6.1 General**

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

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

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

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

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

6.1.6.2 No repeated indication delivery avoidance policy

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

6.1.6.3 Suppress repeated indication delivery avoidance policy

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

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

6.1.6.4 Delayed indication delivery avoidance policy

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

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

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

6.2 Indication filters

6.2.1 General

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

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

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

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

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

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

745 6.2.2 Indication filter coverage

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

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

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

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

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

760 EXAMPLE 1:

```
761 SELECT * FROM CIM_AlertIndication
```

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

766 EXAMPLE 2:

```
767 SELECT * FROM CIM_InstCreation WHERE SourceInstance ISA  
768 CIM_StorageVolume
```

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

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

780 **EXAMPLE 3:**

```
781     SELECT * FROM CIM_AlertIndication WHERE OwningEntity = 'DMTF' AND
782     MessageID = 'SVPC0123'
```

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

788 **EXAMPLE 4:**

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

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

793 **6.2.3 Static indication filters**

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

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

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

- 802 • Indication filters that cover more than one indication

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

- 805 • Indication filters that cover exactly one indication

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

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

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

817 **6.2.4 Indication-specific indication filters**

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

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

824 **6.2.5 Global indication filters**

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

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

832 For details, see 7.3.16.

833 **6.2.6 Dynamic indication filters**

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

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

845 **6.3 Filter collections**

846 **6.3.1 General**

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

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

851 The main purposes of filter collections are:

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

859 **6.3.2 Filter collection coverage**

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

862 **6.3.3 Static filter collections**

863 **6.3.3.1 General**

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

866 **6.3.3.2 Unique identification**

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

870 **6.3.3.3 Defined coverage**

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

878 **6.3.3.4 Profile specific filter collections**

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

884 **6.3.3.5 Global filter collections**

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

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

- 888 • All alert indications
- 889 • All alert indications specified in profiles
- 890 • All lifecycle indications
- 891 • All indications specified in profiles
- 892 • All alert indications specified in profiles
- 893 • All lifecycle indications specified in profiles

894 For details, see 7.3.22.

895 **6.4 Subscriptions, listeners, and listener destinations**

896 **6.4.1 Subscriptions**

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

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

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

911 **6.4.2 Overlapping coverages of subscriptions**

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

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

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

922 **6.4.3 Subscription management authorization**

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

927 **6.4.4 Listeners**

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

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

936 **6.4.5 Listener destinations**

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

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

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

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

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

952 **6.5 Indication service and implementation**

953 **6.5.1 Implementation**

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

957 **6.5.2 Indication service**

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

962 **6.6 Indication system and referencing profiles**

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

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

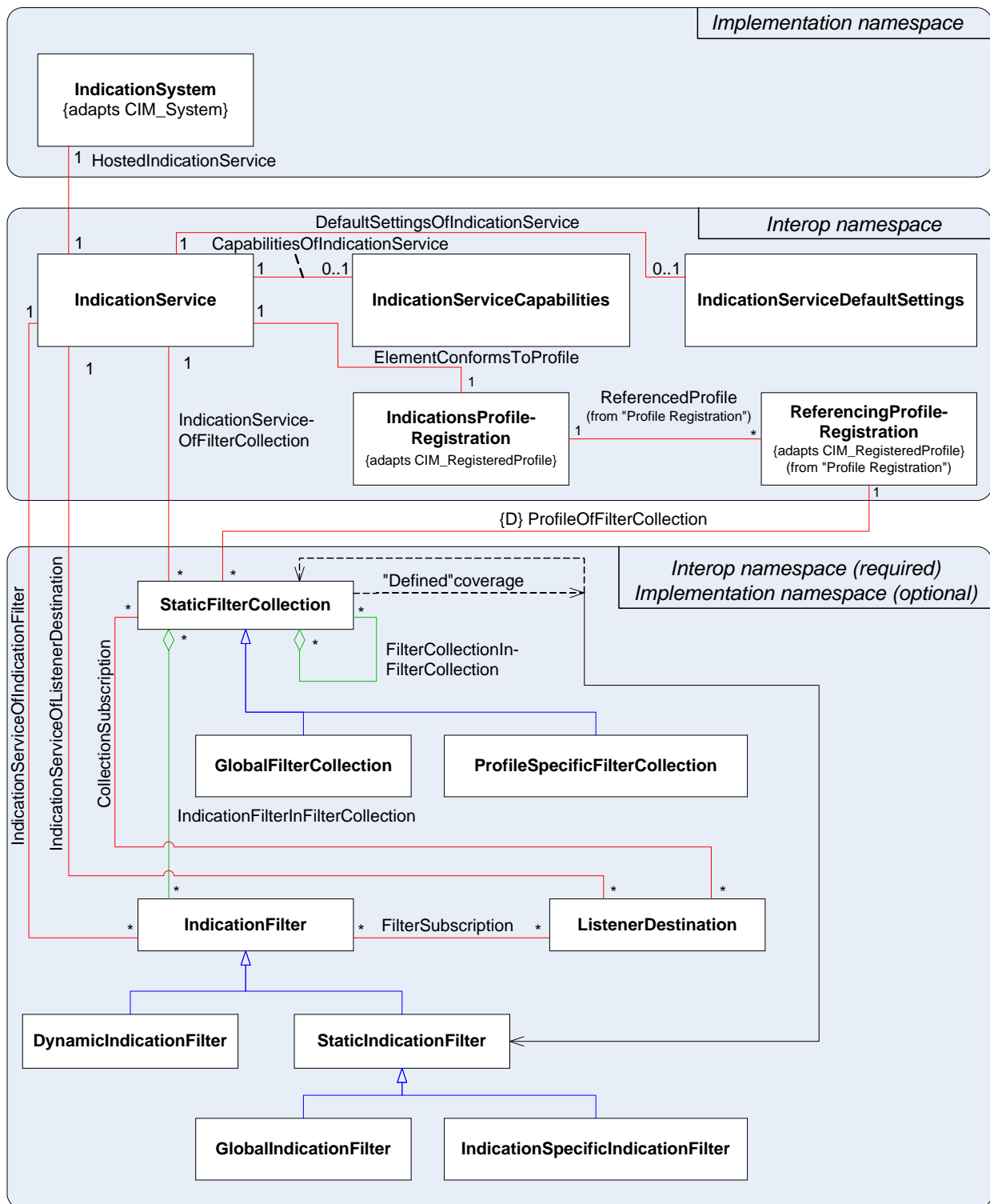
966 In the general case, the scoping systems of referencing profiles are different from the indication system,
967 that is, they are different from the system hosting the WBEM server. In other words, referencing profiles
968 are not required to provide the scope for the indication service, and the central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

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

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

980 **6.7 CIM model**

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



982

983

Figure 2 – Indications Profile: DMTF class adaptation diagram

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

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

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

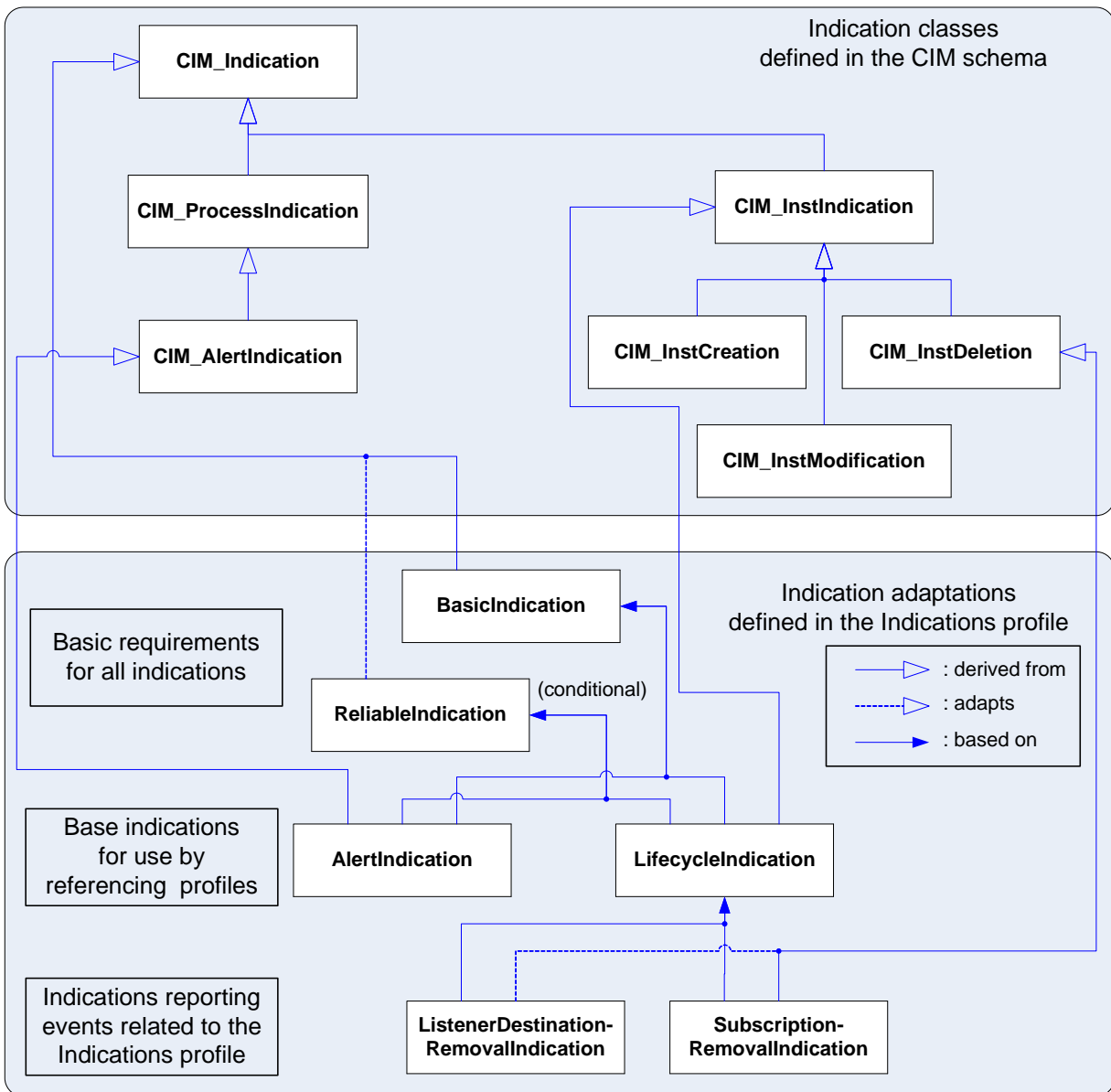
998 **DEPRECATED**

999 The ProfileOfFilterCollection association adaptation models the relationship between filter collections and
1000 the registration of this profile.

1001 NOTE The ProfileOfFilterCollection association adaptation (defined as the CIM_ConcreteDependency "profile
1002 class" in version 1.1 of this profile) is deprecated in version 1.2 of this profile in favor of a naming
1003 convention for static filter collections that enables their unique identification.

1004 **DEPRECATED**

1005 Figure 3 depicts the adaptations of indication classes defined by this profile along with the adapted
 1006 indication classes.



1007

1008 **Figure 3 – Indications Profile: Indication adaptations and adapted indication classes**

1009 The most essential indication adaptations defined in this profile are listed below, along with their modeled
 1010 indications:

- 1011 • the BasicIndication adaptation (see 7.3.29) models indications as described in 6.1.2
- 1012 • the ReliableIndication adaptation (see 7.3.30) models reliable indications as described in 6.1.5;
- 1013 this adaptation specifies additional optional requirements that can be implemented separately
- 1014 from the requirements of other indication adaptations.

- 1015 • the AlertIndication adaptation (see 7.3.31) models alert indication as described in 6.1.2.2; it is
1016 an abstract adaptation available to referencing profiles in order to define their own alert
1017 indications
- 1018 • the LifecycleIndication adaptation (see 7.3.32) models lifecycle indications as described
1019 in 6.1.2.3; it is an abstract adaptation available to referencing profiles in order to define their
1020 own lifecycle indications.

1021 7 Implementation

1022 7.1 Separation of requirements

1023 This profile defines implementation requirements for implementations (for example, WBEM servers
1024 implementing this profile) and for listeners (for example, WBEM listeners implementing this profile).

1025 The implementation requirements for implementations are further separated into WBEM server related
1026 requirements and referencing profile related requirements, as follows:

- 1027 • Requirements that address the infrastructure for the delivery of indications (including the
1028 management of listener destinations and subscriptions) are WBEM server related requirements,
1029 and are typically implemented only once within an implementation.
- 1030 • Requirements that address the generation of indications are related to the referencing profile
1031 defining those indications, and are typically implemented as part of the implementation of that
1032 referencing profile.
- 1033 • Requirements that address functionality related to indication filters and filter collections are
1034 referencing profile related requirements.

1035 However, WBEM servers may contain other facilities allowing implementations of referencing
1036 profiles to delegate some of their implementation responsibilities to these facilities. For example,
1037 within WBEM servers providing a CIM instance repository the implementations of referencing
1038 profiles can delegate storing indication filters and filter collections to the CIM instance
1039 repository, such that in this case the implementation requirements for referencing profiles are
1040 effectively reduced to storing respective objects into the repository when the implementation of
1041 the referencing profile is installed.

1042 In this profile WBEM server related implementation requirements are marked with a phrase such as the
1043 following:

1044 "The requirements in this subclause are WBEM server related implementation requirements."

1045 In this profile referencing profile related implementation requirements are marked with a phrase such as
1046 the following:

1047 "The requirements in this subclause are referencing profile related implementation requirements."

1048 This facilitates explicit distinction of WBEM server related implementation requirements as opposed to
1049 requirements related to the implementation of referencing profiles.

1050 7.2 Features

1051 7.2.1 DynamicIndicationFilters

1052 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1053 The implementation of the DynamicIndicationFilters feature provides functionality for dynamic indication
1054 filters; for a description of dynamic indication filters, see 6.2.6.

1055 The granularity of the DynamicIndicationFilters feature is per IndicationService instance (see 7.3.2).

1056 The requirement level of the DynamicIndicationFilters feature is optional.

1057 The implementation of the DynamicIndicationFilters feature for a particular IndicationService instance is
1058 indicated by a value of True for the FilterCreationEnabled property.

1059 **7.2.2 IndicationServiceInitialSettingsExposed**

1060 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1061 The implementation of the IndicationServiceInitialSettingsExposed feature provides information about the
1062 initial settings of an indication service.

1063 The granularity of the IndicationServiceInitialSettingsExposed feature is per
1064 IndicationService instance (see 7.3.2).

1065 The requirement level of the IndicationServiceInitialSettingsExposed feature is optional.

1066 The availability of the IndicationServiceInitialSettingsExposed feature for a particular IndicationService
1067 instance is indicated by the presence of an IndicationServiceInitialSettings instance (see 7.3.9)
1068 associated through an InitialSettingsOfIndicationService instance (see 7.3.10).

1069 **7.2.3 IndicationServiceModification**

1070 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1071 The implementation of the IndicationServiceModification feature provides functionality for client requested
1072 dynamic modification of an indication service.

1073 The granularity of the IndicationServiceModification feature is per IndicationService instance (see 7.3.2).

1074 The requirement level of the IndicationServiceModification feature is optional.

1075 The availability of the IndicationServiceModification feature for a particular IndicationService instance is
1076 indicated if an IndicationServiceCapabilities (see 7.3.7) instance representing the capabilities of the
1077 represented indication service exists and is associated via the CapabilitiesOfIndicationService association
1078 (see 7.3.8), and in that instance the value True is set for any of the following properties:
1079 FilterCreationEnabledIsSettable, DeliveryRetryAttemptsIsSettable, DeliveryRetryIntervalsSettable,
1080 SubscriptionRemovalActionIsSettable, or SubscriptionRemovalTimeIntervalsSettable.

1081 **7.2.4 ReliableIndications**

1082 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1083 The implementation of the ReliableIndications feature provides functionality for reliable indications as
1084 described in 6.1.5. For further details, see 7.3.30 and 7.4.

1085 The granularity of the ReliableIndications feature is per IndicationService instance (see 7.3.2).

1086 The requirement level of the ReliableIndications feature is optional. The implementation of the
1087 ReliableIndications feature is also optional for listeners; in this case, the granularity is once per listener,
1088 and the discovery mechanism does not apply.

1089 The availability of the ReliableIndications feature for a particular IndicationService instance is indicated by
1090 a value larger than 0 for the DeliveryRetryAttempts property.

1091 **7.2.5 SuppressRepeatNotificationPolicy**

1092 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1093 The implementation of the SuppressRepeatNotificationPolicy feature provides functionality for
1094 suppressing repeated indication delivery by implementing the "suppress repeated indication delivery
1095 avoidance policy", as described in 6.1.6.3.

1096 The granularity of the SuppressRepeatNotificationPolicy feature is per implementation.

1097 The requirement level of the SuppressRepeatNotificationPolicy feature is optional.

1098 The availability of the SuppressRepeatNotificationPolicy feature is indicated by the value 3 (Suppress) for
1099 the RepeatNotificationPolicy property in AbstractSubscription instances (see 7.3.25) representing existing
1100 subscriptions.

1101 NOTE The discovery mechanism specified here is only rudimentary because the feature presence can only be
1102 discovered if at least one exploiting subscription is discovered. A future version of this profile is expected
1103 to introduce a new property into the CIM_IndicationServiceCapabilities class that indicates the presence of
1104 the feature per indication service.

1105 **7.2.6 DelayRepeatNotificationPolicy**

1106 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1107 The implementation of the DelayRepeatNotificationPolicy feature provides functionality for suppressing
1108 repeated indication delivery by implementing the "delayed indication delivery avoidance policy", as
1109 described in 6.1.6.4.

1110 The granularity of the DelayRepeatNotificationPolicy feature is per implementation.

1111 The requirement level of the DelayRepeatNotificationPolicy feature is optional.

1112 The availability of the DelayRepeatNotificationPolicy feature is indicated by the value 4 (Delay) for the
1113 RepeatNotificationPolicy property in AbstractSubscription instances (see 7.3.25) representing existing
1114 subscriptions.

1115 NOTE The discovery mechanism specified here is only rudimentary because the feature presence can only be
1116 discovered if at least one exploiting subscription is discovered. A future version of this profile is expected
1117 to introduce a new property into the CIM_IndicationServiceCapabilities class that indicates the presence of
1118 the feature per indication service.

1119 **7.2.7 IndividualFilterSubscription**

1120 The implementation of the IndividualFilterSubscription feature provides functionality for subscriptions to
1121 individual indication filters.

1122 The granularity of the IndividualFilterSubscription feature is per IndicationFilter instance (see 7.3.11).

1123 The requirement level of the IndividualFilterSubscription feature is optional.

1124 The availability of the IndividualFilterSubscription feature for a particular IndicationFilter instance is
1125 indicated by the value True for the IndividualSubscriptionSupported property.

1126 **7.2.8 FilterCollectionCoverageExposure**

1127 The implementation of the FilterCollectionCoverageExposure feature provides functionality for exposing
1128 the coverage of static filter collections.

1129 The granularity of the FilterCollectionCoverageExposure feature is per
1130 StaticFilterCollection instance (see 7.3.17).

1131 The requirement level of the FilterCollectionCoverageExposure feature is optional.

1132 The availability of the FilterCollectionCoverageExposure feature for a particular StaticFilterCollection
1133 instance is indicated through at least one instance of either the IndicationFilterInFilterCollection
1134 association adaptation (see 7.3.19) or the FilterCollectionInFilterCollection association adaptation (see
1135 7.3.20) referencing the StaticFilterCollection instance.

1136 **7.3 Adaptations**

1137 **7.3.1 Conventions**

1138 This profile repeats the effective values of certain Boolean qualifiers as part of property requirements, or
1139 of method parameter requirements. The following convention is established: If the name of a qualifier is
1140 listed, its effective value is True; if the qualifier name is not listed, its effective value is False. The
1141 convention is applied in the following cases:

- 1142 • In: indicates that the parameter is an input parameter
- 1143 • Out: indicates that the parameter is an output parameter
- 1144 • Key: indicates that the property is a key (that is, its value is part of the instance part)
- 1145 • Required: indicates that the element value shall be non-Null

1146 This profile defines operation requirements based on [DSP0223](#).

1147 For adaptations of ordinary classes and of associations the implementation requirements for operations
1148 are specified in adaptation-specific subclauses of 7.3.

1149 **7.3.2 IndicationService: CIM_IndicationService**

1150 **7.3.2.1 General**

1151 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1152 The IndicationService adaptation models indication services; indication services are described in 6.5.2.

1153 The implementation type of the IndicationService adaptation is: "instantiated".

1154 The IndicationService adaptation shall conform to the requirements for "central classes" defined in the
1155 Profile Registration profile; for details, see [DSP1033](#).

1156 **7.3.2.2 Initial behavior**

1157 If the IndicationServiceInitialSettingsExposed feature (see 7.2.2) is implemented, the initial behavior of an
1158 indication service shall be as exposed by the IndicationServiceInitialSettings instance (see 7.3.9) that is
1159 associated with the IndicationService instance representing that indication service through an
1160 InitialSettingsOfIndicationService instance (see 7.3.10).

1161 If the IndicationServiceInitialSettingsExposed feature (see 7.2.2) is not implemented, then the initial
1162 behavior of the indication service shall be as follows:

- 1163 • Retry the delivery of an indication after a delivery failure three additional times, each time
1164 waiting 20 seconds before the retry, and indicate this behavior with a value of 3 for the
1165 DeliveryRetryAttempts property (see 7.3.2.3.3) and the value 20 for the DeliveryRetryInterval
1166 property (see 7.3.2.3.4) in the IndicationService instance representing the indication service

- 1167 • Remove affected subscriptions after 30 days, and indicate this behavior with a value of 2
 1168 (Remove) for the SubscriptionRemovalAction property (see 7.3.2.3.5), and a value of 2,592,000
 1169 seconds (30 days) for the SubscriptionRemovalTimeInterval property (see 7.3.2.3.6) in the
 1170 IndicationService instance representing the indication service

1171 NOTE With respect to the availability of DynamicIndicationFilters feature (see 7.2.1) as indicated by the value of
 1172 the FilterCreationEnabled property an recommended initial behavior is not established; instead the
 1173 implementation is required to always expose the available behavior; see 7.3.2.3.2.

1174 **7.3.2.3 Element requirements**

1175 **7.3.2.3.1 General**

1176 Table 4 lists the element requirements for the IndicationService adaptation.

1177 **Table 4 – IndicationService: Element requirements**

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

1178 If the ModifyInstance() operation is implemented (see 7.3.2.3.7), the values of some properties might be
 1179 modifiable through client requests; see 7.3.7 for details on indicating those properties whose values are
 1180 actually modifiable.

1181 **7.3.2.3.2 Property: FilterCreationEnabled**

1182 The value of the FilterCreationEnabled property shall reflect whether the DynamicIndicationFilters feature
 1183 (see 7.2.1) is available for the IndicationService instance. A value of False indicates that the feature is not
 1184 available; a value of True indicates that the feature is available.

1185 **7.3.2.3.3 Property: DeliveryRetryAttempts**

1186 The value of the DeliveryRetryAttempts property shall reflect the number of times that the implementation
 1187 is going to retry the delivery of an indication to a particular listener in the case of delivery failures. This
 1188 value does not include the initial delivery attempt.

1189 A value larger than 0 indicates that the ReliableIndications feature (see 7.2.4) is available. The value 0
 1190 indicates that the ReliableIndications feature is not available.

1191 **7.3.2.3.4 Property: DeliveryRetryInterval**

1192 The value of the DeliveryRetryInterval property shall reflect the minimal time interval in seconds that the
 1193 implementation waits before delivering an indication to a particular listener destination after a previous
 1194 delivery failure.

1195 **7.3.2.3.5 Property: SubscriptionRemovalAction**

1196 The value of the SubscriptionRemovalAction property shall reflect the removal action for subscriptions
 1197 after two failed indication deliveries where the time interval between the failed deliveries, without any
 1198 intermediate successful indication delivery, exceeds the timeout reflected by the value of the
 1199 SubscriptionRemovalTimeInterval property.

1200 **7.3.2.3.6 Property: SubscriptionRemovalTimeInterval**

1201 The value of the SubscriptionRemovalTimeInterval property shall reflect the minimum time interval that
 1202 implementations shall wait after two failed indication deliveries without any intermediate successful
 1203 indication delivery, before performing the activity designated by the value of the
 1204 SubscriptionRemovalAction property.

1205 **7.3.2.3.7 Method: ModifyInstance()**

1206 The implementation of the ModifyInstance() operation enables clients to modify aspects of the behavior
 1207 of the represented indication service.

1208 The requirement level of the ModifyInstance() operation is conditional.

1209 Condition: The IndicationServiceModification feature is implemented; for a description, see 7.2.3.

1210 Information about which properties are modifiable is provided by an IndicationServiceCapabilities
 1211 instance that is associated to the IndicationService instance representing the indication service; see 7.3.7
 1212 and 7.3.8.

1213 Table 5 lists the error reporting requirements for the ModifyInstance() operation on IndicationService
 1214 instances. If any of the error situations described in the Description column of Table 5 matches, the
 1215 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1216 reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1217 **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.

1218 If the ModifyInstance() operation is successful, the requested modification on the indication service shall
 1219 be applied, and — as a consequence — shall be reflected in all IndicationService instances that
 1220 represent the modified indication service and are exposed by the implementation.

1221 If the ModifyInstance() operation fails, the requested modification on the indication service shall not be
 1222 applied, and — as a consequence — all IndicationService instances that represent the indication service
 1223 shall remain unchanged.

1224 **7.3.2.4 Instance requirements**

1225 Within an implementation there shall be exactly one indication service. That indication service shall be
 1226 represented by an IndicationService instance in the Interop namespace.

1227 NOTE 1 The reasons for requiring exactly one indication service are a) other elements defined in this profile (such
 1228 as subscriptions, listener destinations, or dynamic indication filters) require a relationship to the indication
 1229 service, and b) the modeled use of the CreateInstance() operation does not provide for expressing that
 1230 required relationship at creation time. For these reasons an indication service must be implied at creation
 1231 time, and the simplest approach for that is allowing just one indication service. Future versions of this
 1232 profile might lift the single instance restriction, for example by modeling respective creation methods with
 1233 parameters that enable establishing the required relationship to a specifiable indication service.

1234 NOTE 2 In some places in this profile multiple indication services are mentioned. This is not meant to lift the
 1235 restriction established in this subclause, but to accommodate the future introduction of multiple indication
 1236 services.

1237 **7.3.3 IndicationSystem: CIM_System**

1238 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1239 The IndicationSystem adaptation models indication systems; indication systems are described in 6.6.

1240 The implementation type of the IndicationSystem adaptation is: "instantiated".

1241 The IndicationSystem adaptation shall conform to the requirements for "scoping classes" defined in the
 1242 Profile Registration profile; for details, see [DSP1033](#).

1243 Table 6 lists the element requirements of the IndicationSystem adaptation.

1244 **Table 6 – IndicationSystem: Element requirements**

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

1245 **7.3.4 HostedIndicationService: CIM_HostedService**

1246 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1247 The HostedIndicationService adaptation models the relationship between an indication service and its
 1248 hosting indication system.

1249 The implementation type of the HostedIndicationService association adaptation is: "instantiated".

1250 Table 7 lists the element requirements for the HostedIndicationService association adaptation.

1251 **Table 7 – HostedIndicationService: Element requirements**

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

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

1252 Each IndicationSystem instance (see 7.3.3) shall be associated through a HostedIndicationService
 1253 instance with the IndicationService instance (see 7.3.2) representing the indication service hosted by the
 1254 indication system represented by the IndicationSystem instance.

1255 **7.3.5 IndicationsProfileRegistration: CIM_RegisteredProfile**

1256 **7.3.5.1 General**

1257 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1258 The IndicationsProfileRegistration adaptation models the profile registration of this profile, that is, the
 1259 representation of the specific implemented version 1.2.0 of this profile.

1260 The implementation type of the IndicationsProfileRegistration adaptation is: "instantiated".

1261 The specific implemented version of this profile shall be represented by IndicationsProfileRegistration
 1262 instances in the Interop namespace.

1263 NOTE The existence of an instance of this adaptation indicates that version 1.2.0 of this profile is implemented at
 1264 least once within the WBEM server.

1265 Table 8 lists the element requirements for the IndicationsProfileRegistration adaptation.

1266 **Table 8 – IndicationsProfileRegistration: Element requirements**

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

1267 NOTE Operation requirements are defined by the base "profile class" CIM_RegisteredProfile defined in
 1268 [DSP1033](#).

1269 **7.3.6 ElementConformsToProfile: CIM_ElementConformsToProfile**

1270 The ElementConformsToProfile adaptation models the relationship between an indication service and the
 1271 profile registration of this profile (see 7.3.5).

1272 The implementation type of the ElementConformsToProfile association adaptation is: "instantiated".

1273 Table 9 lists the element requirements for the ElementConformsToProfile association adaptation.

1274 **Table 9 – ElementConformsToProfile: Element requirements**

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

1275 Each IndicationService instance (see 7.3.2) shall be associated through an ElementConformsToProfile
1276 instance with an IndicationsProfileRegistration instance (see 7.3.5).

1277 NOTE By requiring the implementation of the ElementConformsToProfile adaptation, this profile in fact requires
1278 the central class profile advertisement methodology defined in [DSP1033](#). The scoping class profile
1279 advertisement methodology is not applicable because the central instances of implementations of
1280 referencing profiles will in almost all cases not be identical with the central instance of this profile, that is,
1281 the IndicationSystem instance required by 7.3.3. Note that this does not restrict referencing profiles from
1282 choosing a different methodology for their profile advertisement.

1283 **7.3.7 IndicationServiceCapabilities: CIM_IndicationServiceCapabilities**

1284 **7.3.7.1 General**

1285 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1286 The IndicationServiceCapabilities adaptation models the capabilities of indication services; indication
1287 services are described in 6.5.2.

1288 The requirement level of the IndicationServiceCapabilities adaptation is conditional.

1289 Condition: The IndicationServiceModification feature is implemented; see 7.2.3.

1290 The implementation type of the IndicationServiceCapabilities adaptation is: "instantiated".

1291 **7.3.7.2 Element requirements**

1292 **7.3.7.2.1 General**

1293 Table 10 lists the element requirements for the IndicationServiceCapabilities adaptation.

1294 **Table 10 – IndicationServiceCapabilities: Element requirements**

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

1295 **7.3.7.2.2 Property: FilterCreationEnabledIsSettable**

1296 **DEPRECATED**

1297 The value of the FilterCreationEnabledIsSettable property shall indicate whether the implementation
 1298 supports modification of the FilterCreationEnabled property of the associated IndicationService instance.

1299 NOTE Values other than False are deprecated because it does not make sense enabling clients to set values of
 1300 properties that represent functionality that is either implemented or not implemented.

1301 **DEPRECATED**

1302 The value of the FilterCreationEnabledIsSettable property should be False, indicating that the
 1303 implementation does not support the modification of the FilterCreationEnabled property of the associated
 1304 IndicationService instance.

1305 **7.3.8 CapabilitiesOfIndicationService: CIM_ElementCapabilities**

1306 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.
 1307 The CapabilitiesOfIndicationService adaptation models the relationship between an indication service and
 1308 its capabilities.
 1309 The requirement level of the CapabilitiesOfIndicationService adaptation is conditional.
 1310 Condition: The IndicationServiceModification feature is implemented; see 7.2.3.
 1311 The implementation type of the CapabilitiesOfIndicationService association adaptation is: "instantiated".
 1312 Table 11 lists the element requirements for the CapabilitiesOfIndicationService association adaptation.

1313 **Table 11 – CapabilitiesOfIndicationService: Element requirements**

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

1314 Each IndicationService instance (see 7.3.2) shall be associated through a CapabilitiesOfIndicationService
 1315 instance with at most one IndicationServiceCapabilities instance (see 7.3.7) representing the capabilities
 1316 of the indication service represented by the IndicationService instance.

1317 **7.3.9 IndicationServiceInitialSettings: CIM_IndicationServiceSettingData**

1318 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.
 1319 The IndicationServiceInitialSettings adaptation models initial settings for indication services; indication
 1320 services are described in 6.5.2. The initial settings of an indication service are the settings that apply at
 1321 the point in time when the WBEM server hosting the indication service initially starts up the indication
 1322 service.
 1323 The requirement level of the IndicationServiceInitialSettings adaptation is conditional.
 1324 Condition: The IndicationServiceInitialSettingsExposed feature is implemented; see 7.2.2.
 1325 The implementation type of the IndicationServiceInitialSettings adaptation is: "instantiated".
 1326 Table 12 lists the element requirements for the IndicationServiceInitialSettings adaptation.

1327

Table 12 – IndicationServiceInitialSettings: Element requirements

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

1328 The initial settings of an indication service shall be represented by an IndicationServiceInitialSettings
 1329 instance in the Interop namespace.

1330 **7.3.10 InitialSettingsOfIndicationService: CIM_ElementSettingData**

1331 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1332 The InitialSettingsOfIndicationService association adaptation models the relationship between an
 1333 indication service and its initial settings; indication services are described in 6.5.2.

1334 The requirement level of the InitialSettingsOfIndicationService association adaptation is conditional.

1335 Condition: The IndicationServiceInitialSettingsExposed feature is implemented; see 7.2.2.

1336 The implementation type of the InitialSettingsOfIndicationService association adaptation is: "instantiated".

1337 Table 13 lists the element requirements for the InitialSettingsOfIndicationService association adaptation.

1338

Table 13 – InitialSettingsOfIndicationService: Element requirements

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

1339 Each IndicationService instance (see 7.3.2) shall be associated through a
 1340 InitialSettingsOfIndicationService instance with at most one IndicationServiceInitialSettings instance (see
 1341 7.3.9) representing the initial settings of the indication service represented by the IndicationService
 1342 instance.

1343 **7.3.11 IndicationFilter: CIM_IndicationFilter**

1344 **7.3.11.1 General**

1345 The requirements in this subclause are referencing profile and WBEM server related implementation
 1346 requirements.

1347 The IndicationFilter adaptation models indication filters; indication filters are described in 6.2.

1348 The implementation type of the IndicationFilter adaptation is: "abstract".

1349 **7.3.11.2 Semantical requirements**

1350 For a particular indication filter the implementation shall filter any indication generated by (indication-
 1351 specific parts of) the implementation that is within the coverage of the indication filter, that is, that meets
 1352 both of the following requirements:

- 1353 • it matches the query statement (see 7.3.11.3.5) given by the value of the Query property in the
 1354 IndicationFilter instance representing the indication filter
- 1355 • its indication origin (see 6.1.2.4) is one of the local namespaces identified by the value of the
 1356 SourceNamespaces[] array property in that instance, or, in case that value is NULL, is the local
 1357 namespace in which the IndicationFilter instance representing the indication filter resides

1358 For the particular indication filter the implementation shall ignore any generated indication that does not
 1359 meet these requirements.

1360 Indications that passed an indication filter need to be further processed; see the requirements on the
 1361 IndicationFilterName property defined in 7.3.29.4.2, and the semantical requirements on listener
 1362 destinations defined in 7.3.23.2, and on subscriptions defined in 7.3.25.2. If implemented, the
 1363 requirements for reliable indications as defined in 7.3.30 and 7.4 may apply.

1364 Note that the indication filter semantics apply regardless of which profile specified the indications and
 1365 indication filters; thus an indication specified in one referencing profile is required to be considered by
 1366 indication filters specified in that referencing profile, but also by those specified in any other referencing
 1367 profile or in this profile and by those not specified in any profile.

1368 The indication filter semantics defined in this subclause do not require that an implementation implements
 1369 any of the indications within the coverage of an indication filter. However, referencing profiles may define
 1370 additional semantics for indication filters they define, including the case that the existence of a particular
 1371 IndicationFilter instance indicates that one or all indications within the coverage of the represented
 1372 indication filter are implemented. Of course, this approach is only feasible if the coverage covers one or
 1373 just a few indications.

1374 **7.3.11.3 Element requirements**

1375 **7.3.11.3.1 General**

1376 Table 14 lists the element requirements for the IndicationFilter adaptation.

1377 **Table 14 – IndicationFilter: Element requirements**

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

1378 **7.3.11.3.2 Property: Name**

1379 The value of the Name property shall be the name of the indication filter; it shall be formatted as defined
 1380 by the following ABNF rule:

1381 `OrgID ":" RegisteredName ":" UniqueID`

1382 `OrgID` shall identify the business entity owning the referencing profile. `OrgID` shall include a copyrighted,
 1383 trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID
 1384 assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness,
 1385 `OrgID` shall not contain a colon (:). For referencing profiles owned by DMTF, `OrgID` shall match
 1386 "DMTF".

1387 RegisteredName shall be the registered name of the referencing profile, as defined by the value of the
1388 RegisteredName property in the RegisteredProfile instance representing the implemented version of that
1389 profile.

1390 UniqueID shall uniquely identify the represented indication filter within the referencing profile.

1391 **DEPRECATED**

1392 For compatibility with version 1.0 of this profile, referencing profiles owned by business entities other than
1393 DMTF may in addition define values for the Name property that are formatted as defined by the following
1394 ABNF rule:

1395 OrgID ":" UniqueID

1396 Where:

1397 OrgID is defined above in this subclause.

1398 UniqueID shall uniquely identify the instance within the business entity owning the referencing
1399 profile.

1400 Version 1.1 of this profile has deprecated this additional format.

1401 **DEPRECATED**

1402 **7.3.11.3.3 Property: SourceNamespaces**

1403 A non-Null value of this property is required for IndicationFilter instances in the Interop namespace; for
1404 IndicationFilter instances in other namespaces it is optional.

1405 If not Null, the value of the SourceNamespaces[] array property shall contain the names of local
1406 namespaces that are considered as potential indication origin namespaces (see 6.1.2.4) during indication
1407 filtering; see 7.3.11.2. The value shall not be an empty array.

1408 It is not required that the local namespaces identified by elements of value of the SourceNamespaces[]
1409 array property exist. If a non-existing local namespace is identified, no indications can originate out of that
1410 non-existing namespace; consequently, that element does not have an effect on indication filtering.
1411 However, if the identified namespace is added to the implementation at a later point in time, per the
1412 requirements of 7.3.11.2 indications originating out of that namespace are to be considered for indication
1413 filtering from then on.

1414 The value elements of the SourceNamespaces[] array property shall be formatted using the format that
1415 the implementation uses for value of the Name property in instances of the CIM_Namespace class that
1416 represent namespaces.

1417 **7.3.11.3.4 Property: IndividualSubscriptionSupported**

1418 The value of the IndividualSubscriptionSupported property shall be True if the IndividualFilterSubscription
1419 feature (see 7.2.7) is available for the IndicationFilter instance; otherwise, the value shall be False.

1420 **7.3.11.3.5 Property: Query**

1421 The value of the Query property shall be a properly formed query statement that is conformant to the
1422 requirements of the query language identified by the value of the QueryLanguage property, and that
1423 states the coverage of the indication filter.

1424 **7.3.11.3.6 Property: QueryLanguage**

1425 The value of the QueryLanguage property shall identify the query language in which the query statement
 1426 exposed by the value of the Query property is expressed.

1427 NOTE This profile presently does not define a straight forward mechanism enabling clients to discover the set of
 1428 query languages supported by an implementation. A future version of this profile is expected to introduce
 1429 such a mechanism. For now, a rudimentary workaround may be inspecting the CIM representation of
 1430 existing indication filters, thereby discovery a lower boundary for the set of supported query languages.

1431 **7.3.11.4 Instance requirements**

1432 Indication filters (see 6.2) shall be represented by IndicationFilter instances in the Interop namespace.

1433 The representation in namespaces other than the Interop namespace should be avoided. However, if
 1434 additional IndicationFilter instances represent an indication filter also in implementation namespaces,
 1435 these instances shall have the same key property values as the one in the Interop namespace.

1436 **7.3.12 StaticIndicationFilter: CIM_IndicationFilter**

1437 **7.3.12.1 General**

1438 The requirements in this subclause are referencing profile and WBEM server related implementation
 1439 requirements.

1440 The StaticIndicationFilter adaptation models static indication filters; static indication filters are described in
 1441 6.2.3.

1442 The implementation type of the StaticIndicationFilter adaptation is: "abstract".

1443 **7.3.12.2 Element requirements**

1444 **7.3.12.2.1 General**

1445 Table 15 lists the element requirements for the StaticIndicationFilter adaptation.

1446 **Table 15 – StaticIndicationFilter: Element requirements**

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

1447 **7.3.12.2.2 Property: QueryLanguage**

1448 In adaptations based on the StaticIndicationFilter adaptation in referencing profiles owned by DMTF, the
 1449 value shall be "CQL", thereby requiring CQL as the query language.

1450 **7.3.13 DynamicIndicationFilter: CIM_IndicationFilter**

1451 **7.3.13.1 General**

1452 The requirements in this subclause are WBEM server related implementation requirements.

1453 The DynamicIndicationFilter adaptation models dynamic indication filters; dynamic indication filters are
1454 described in 6.2.6.

1455 The requirement level of the DynamicIndicationFilter adaptation is conditional.

1456 Condition: The DynamicIndicationFilters feature is implemented; see 7.2.1.

1457 The implementation type of the DynamicIndicationFilter adaptation is: "instantiated".

1458 **7.3.13.2 Element requirements**

1459 **7.3.13.2.1 General**

1460 Table 16 lists the element requirements for the DynamicIndicationFilter adaptation.

1461 **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.

1462 **7.3.13.2.2 Operation: CreateInstance()**

1463 Table 17 lists the error reporting requirements for the CreateInstance() operation on
1464 DynamicIndicationFilter instances. If any of the error situations described in the Description column of
1465 Table 17 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
1466 addition, the error reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

1467 **Table 17 – CreateInstance(): Error reporting requirements**

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

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

1468 If the CreateInstance() operation is successful, the requested dynamic indication filter shall be created,
 1469 and — as a consequence — shall be represented by a DynamicIndicationFilter instance in the requested
 1470 namespace.

1471 Clients should abstain from requesting the creation of DynamicIndicationFilter instances in namespaces
 1472 other than the Interop namespace. However, if the requested namespace is not the Interop namespace,
 1473 the implementation shall expose an additional DynamicIndicationFilter instance representing the dynamic
 1474 indication filter in the Interop namespace. That instance shall have identical values for all properties
 1475 except for the SourceNamespaces[] array property for which the provisions of 7.3.11.3.3 apply.

1476 If the CreateInstance() operation is fails, no dynamic indication filter shall be created, and — as a
 1477 consequence — no representing DynamicIndicationFilter instances shall be exposed in any namespace.

1478 **DEPRECATED**

1479 If the returned CIM status code is CIM_ERR_FAILED because an indication filter with the same coverage
 1480 as that requested already exists, the object path of the CIM_IndicationFilter instance representing the
 1481 existing indication filter in the Interop namespace shall be returned as the value of the ErrorSource
 1482 property in the CIM_Error instance accompanying the CIM status code.

1483 NOTE Only this specific ad-hoc use of CIM_Error is deprecated. It is intended that a future version of this profile
 1484 introduces extended error handling based on standard error messages.

1485 **DEPRECATED**

1486 With respect to input values for key properties the rules defined in [DSP1001](#) apply, namely that
 1487 implementation may ignore any input value for non-reference key properties, and that clients should
 1488 abstain from providing input values for key properties.

1489 **7.3.13.2.3 Operation: DeleteInstance()**

1490 Table 18 lists the error reporting requirements for the DeleteInstance() operation on
 1491 DynamicIndicationFilter instances, and related CIM status codes. If any of the error situations described
 1492 in the Description column of Table 18 matches, the operation shall fail and the corresponding CIM status
 1493 code shall be returned. In addition, the error reporting requirements defined in [DSP0223](#) for the
 1494 DeleteInstance() operation apply.

1495

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).

1496 If the DeleteInstance() operation succeeds, the represented dynamic indication filter shall be deleted and
 1497 — as a consequence — no longer be represented by any DynamicIndicationFilter instances in any
 1498 namespace exposed by the implementation.

1499 NOTE The instance requirements of associations representing relationships of the deleted dynamic indication
 1500 filter imply that respective association instances in any namespace exposed by the implementation cease
 1501 to exist; in this case this applies to IndicationServiceOfIndicationFilter instances (see 7.3.14). However,
 1502 note that the DeleteInstance() operation for the dynamic indication filter is required to fail if subscriptions
 1503 exist.

1504 If the DeleteInstance() operation fails, the dynamic indication filter shall not be deleted, and — as a
 1505 consequence — any representing DynamicIndicationFilter instances shall continue to exist as before.

1506 **7.3.13.2.4 Operation: ModifyInstance()**

1507 The implementation of the ModifyInstance() operation enables clients to modify aspects of the behavior
 1508 of the represented indication filter.

1509 The requirement level of the ModifyInstance() operation is optional.

1510 Table 19 lists the error reporting requirements for the ModifyInstance() operation on
 1511 DynamicIndicationFilter instances. If any of the error situations described in the Description column of
 1512 Table 19 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1513 addition, the error reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1514

Table 19 – ModifyInstance(): Error reporting requirements

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

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

1515 If the ModifyInstance() operation is successful, the requested modification on the dynamic indication filter
 1516 shall be applied, and — as a consequence — shall be reflected in all DynamicIndicationFilter instances
 1517 that represent the modified dynamic indication filter and are exposed by the implementation.

1518 If the ModifyInstance() operation is fails, the requested modification on the dynamic indication filter shall
 1519 not be applied, and — as a consequence — all DynamicIndicationFilter instances that represent the
 1520 dynamic indication filter shall remain unchanged.

1521 **7.3.13.3 Instance requirements**

1522 Dynamic indication filters shall be represented by DynamicIndicationFilter instances; the additional
 1523 requirements of 7.3.11.4 apply.

1524 **7.3.14 IndicationServiceOfIndicationFilter: CIM_ServiceAffectsElement**

1525 The requirements in this subclause are referencing profile and WBEM server related implementation
 1526 requirements.

1527 The IndicationServiceOfIndicationFilter adaptation models the relationship between indication services
 1528 and the indication filters they manage.

1529 The implementation type of the IndicationServiceOfIndicationFilter association adaptation is:
 1530 "instantiated".

1531 Table 20 lists the element requirements for the IndicationServiceOfIndicationFilter association adaptation.

1532 **Table 20 – IndicationServiceOfIndicationFilter: Element requirements**

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

1533 Each IndicationService instance (see 7.3.2) shall be associated through an
 1534 IndicationServiceOfIndicationFilter instance with each IndicationFilter instance (see 7.3.11) representing
 1535 an indication filter managed by the indication service represented by the IndicationService instance.

1536 **7.3.15 IndicationSpecificIndicationFilter: CIM_IndicationFilter**

1537 **7.3.15.1 General**

1538 The requirements in this subclause are referencing profile and WBEM server related implementation
 1539 requirements.

1540 The IndicationSpecificIndicationFilter adaptation models indication-specific indication filters for indications
 1541 defined in referencing profiles or in this profile; indication-specific indication filters are described in 6.2.4.

1542 The requirement level of the IndicationSpecificIndicationFilter adaptation is optional.

1543 The IndicationSpecificIndicationFilter adaptation should be implemented if indications defined in a
 1544 referencing profile or in this profile are implemented.

1545 The implementation type of the IndicationSpecificIndicationFilter adaptation is: "instantiated".

1546 **7.3.15.2 Element requirements**

1547 **7.3.15.2.1 General**

1548 Table 21 lists the element requirements for the IndicationSpecificIndicationFilter adaptation.

1549 **Table 21 – IndicationSpecificIndicationFilter: Element requirements**

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

1550 **7.3.15.2.2 Property: Name**

1551 The value of the Name property shall be formatted as defined by the following ABNF rule:

```
1552   OrgID ":" RegisteredName ":" IndicationAdaptationName "Filter" [ "/"
1553   MessageIdentification ]
```

1554 OrgID and RegisteredName shall be specified as detailed in 7.3.11.3.2.

1555 IndicationAdaptationName shall be the name of the indication adaptation defined in the profile
 1556 identified by the RegisteredName rule. If the indication adaptation defines more than one possible
 1557 indication.

1558 The MessageIdentification suffix only applies for the representation of indication-specific indication
 1559 filters covering alert indications modeled by an adaptation based on the AlertIndication adaptation (see
 1560 7.3.31); in this case for each alert indication defined by an alert message reference in the profile, a
 1561 specific IndicationSpecificIndicationFilter instance is defined, where MessageIdentification shall be
 1562 set as defined in 7.3.31.2 for the CIM representation of the alert indication. Thus, for alert indications,

1563 there is a one-to-one relationship between defined referenced alert messages and possible
1564 corresponding IndicationSpecificIndicationFilter instances.

1565 For lifecycle indications the suffix is not necessary because adaptations based on the LifecycleIndication
1566 adaptation (see 7.3.32) only can address one event, as defined by a (constant) query statement. Thus,
1567 for lifecycle indications, there is a one-to-one relationship between defined lifecycle indications and
1568 possible corresponding IndicationSpecificIndicationFilter instances.

1569 7.3.15.2.3 Property: Query

1570 The value of the Query property shall be identical with the event definition query statement (see 7.3.29.2)
1571 of the indication adaptation defined in the referencing profile or in this profile that is covered by the
1572 represented indication-specific indication filter. In the case IndicationSpecificIndicationFilter instances
1573 covering alert indications modeled by an adaptation based on the AlertIndication adaptation, the value of
1574 the Query property shall apply the ABNF rule named `EventQuerySingle` (see 7.3.31.2); that way for
1575 alert indication adaptation referencing more than one alert message, separate
1576 IndicationSpecificIndicationFilter instances are defined for each referenced alert message.

1577 7.3.15.3 Instance requirements

1578 If a profile defines an indication adaptation based on the AlertIndication adaptation (see 7.3.31) or the
1579 Lifecycle adaptation (see 7.3.32), a corresponding indication-specific indication filter may be represented
1580 by an IndicationSpecificIndicationFilter instance, with respective values of the Name and Query
1581 properties.

1582 NOTE As with any indication filter (see 6.2.1), the existence of an indication-specific indication filter and its
1583 representation by an IndicationSpecificIndicationFilter instance does not imply that the covered indication
1584 is actually implemented. Furthermore, in the case where multiple implementations of the referencing profile
1585 exist in a WBEM server, multiple IndicationSpecificIndicationFilter instances with identical values for Name
1586 and Query properties may result.

1587 This profile leaves the decision whether or not to represent indication-specific indication filters as
1588 IndicationSpecificIndicationFilter instances to the implementation; however, referencing profiles can
1589 define an adaptation based on IndicationSpecificIndicationFilter adaptation that state more strict instance
1590 requirements.

1591 In any case, if an implementation decides to represent indication-specific indication filters, these are to be
1592 represented as required by the IndicationSpecificIndicationFilter adaptation. In addition, the requirements
1593 of related adaptations such as the ProfileSpecificFilterCollection adaptation (see 7.3.21) or the
1594 IndicationFilterInFilterCollection associations adaptation (see 7.3.19) apply.

1595 7.3.16 GlobalIndicationFilter: CIM_IndicationFilter

1596 7.3.16.1 General

1597 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

1598 The GlobalIndicationFilter adaptation models a global indication filters; global indication filters are
1599 described in 6.2.5.

1600 The implementation type of the GlobalIndicationFilter adaptation is: "instantiated".

1601 7.3.16.2 Element requirements

1602 Table 22 lists the element requirements for the GlobalIndicationFilter adaptation.

1603

Table 22 – GlobalIndicationFilter: Element requirements

Element	Requirement	Description
Base adaptations		
StaticIndicationFilter	Mandatory	See 7.3.12.

1604

7.3.16.3 Instance requirements

1605

7.3.16.3.1 Instance requirements related to alert indications

1606

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

1607

1608

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

1609

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

1610

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

1611

1612

7.3.16.3.2 Instance requirements related to lifecycle indications

1613

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

1614

1615

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

1616

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"

1617

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

1618

1619

1620

7.3.17 StaticFilterCollection: CIM_FilterCollection

1621

7.3.17.1 General

1622

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

1623

1624

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

1625

1626

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

1627 7.3.17.2 Semantical requirements

1628 The coverage of a filter collection shall be the aggregated coverage of all the indication gates contained
1629 by the filter collection. This definition applies recursively to contained filter collections.

1630 NOTE Since filter collections aggregate the coverages of contained indication filters and contained other filter
1631 collections, and do not specify a filter query statement on their own, the defined coverage of a static filter
1632 collection is finally described by the set of query statements of its (directly or indirectly) aggregated
1633 indication filters.

1634 The implementation shall filter all indications generated by (indication-specific parts of) the
1635 implementation that are within the coverage of a filter collection.

1636 The implementation shall ignore any generated indication that is outside the coverage of the filter
1637 collection.

1638 If a particular indication is within the coverage of more than one indication gate contained by a filter
1639 collection, that indication shall pass the filter collection only once, and shall not be replicated for every
1640 matching contained indication gate.

1641 Indications that passed a filter collection need to be further processed; see the requirements on the
1642 IndicationFilterName property defined in 7.3.29.4.2, and the semantical requirements on listener
1643 destinations defined in 7.3.23.2, and on subscriptions defined in 7.3.25.2. If implemented, the
1644 requirements for reliable indications as defined in 7.3.30 and 7.4 may apply.

1645 These semantics apply regardless of whether all, some or no contained indication gates are represented
1646 as collection members in CIM. Thus clients and listeners need to be aware of the fact that the coverage of
1647 a static filter collection may be larger than that observable through inspection of CIM represented
1648 members of that static filter collection. In other words, indications could be delivered to subscribed
1649 listeners that are within the coverage of members of the static filter collection that are not currently
1650 represented in CIM; in the extreme case no members at all are CIM represented. On the other hand,
1651 even if the coverage of a static filter collection is not represented through CIM, clients may have a priori
1652 knowledge about the defined coverage of that static filter collection, for example by means of built-in
1653 program code or data; see 7.3.17.3.

1654 NOTE During runtime, the set of members of a static filter collection and the extent to which such members are
1655 represented in CIM may change. For example, consider the global filter collection with a defined coverage
1656 covering all alert indications defined in referencing profiles, as defined in 7.3.22.4.1. Its member set might
1657 grow or shrink over time as implementations of referencing profiles are installed in or removed from the
1658 implementation; however, the conceptual defined coverage of "all alert indications defined in referencing
1659 profile" remains constant.

1660 7.3.17.3 Requirements pertaining to the defined coverage

1661 For concrete adaptations based (directly or indirectly) on the StaticFilterCollection adaptation, profiles
1662 shall specify a defined coverage (see 6.3.3.3) through normative text that identifies indication filters
1663 and/or other filter collections as the *contained members* of the static filter collection, and thereby —
1664 because of 7.3.17.2 — as contributors to the coverage of the static filter collection.

1665 NOTE If in a chain of (abstract and concrete) adaptations based on the StaticFilterCollection adaptation the
1666 defined coverage is defined as part of an intermediate (abstract or concrete) adaptation, that definition
1667 propagates into adaptations (directly or indirectly) based on that intermediate adaptation.

1668 The defined coverage of a static filter collection always applies regardless of whether any members are
1669 represented in CIM. For contained static filter collections the specification of a defined coverage is
1670 likewise required.

1671 The definition of the defined coverage may be specified at the level of adaptations, or may be broken
1672 down to individual adaptation instances, or both.

1673 For examples of how to specify a defined coverage, see 7.3.21.3 and 7.3.22.

1674 **7.3.17.4 Element requirements**

1675 **7.3.17.4.1 General**

1676 Table 25 lists the element requirements for the StaticFilterCollection adaptation.

1677 **Table 25 – StaticFilterCollection: Element requirements**

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

1678 **7.3.17.4.2 Property: CollectionName**

1679 The value of the CollectionName property shall be formatted as defined by the following ABNF rule:

1680 `OrgID ":" RegisteredName ":" UniqueID`

1681 OrgID shall identify the business entity owning the referencing profile. OrgID shall include a copyrighted, trademarked, or otherwise unique name that is owned by that business entity or that is a registered ID assigned to that business entity by a recognized global authority. In addition, to ensure uniqueness, 1682 1683 1684 OrgID shall not contain a colon (:).

1685 For referencing profiles owned by DMTF, OrgID shall match "DMTF".

1686 RegisteredName shall be the registered name of the referencing profile, as defined by the value of the 1687 RegisteredName property in the RegisteredProfile instance representing the implemented version of the 1688 referencing profile.

1689 UniqueID shall uniquely identify the instance within the implementation of the referencing profile.

1690 **DEPRECATED**

1691 For compatibility with version 1.0 of this profile, referencing profiles owned by business entities other than 1692 DMTF may in addition define values for the Name property that are formatted as defined by the following 1693 ABNF rule:

1694 `OrgID ":" UniqueID`

1695 Where:

- 1696 OrgID is defined above in this subclause.
- 1697 UniqueID shall uniquely identify the instance within the business entity owning the referencing
- 1698 profile.
- 1699 Version 1.1 of this profile has deprecated this additional format.

1700 **DEPRECATED**

1701 **7.3.17.5 Instance requirements**

1702 Static filter collections (see 6.3.3) shall be represented by StaticFilterCollection instances in the Interop

1703 namespace.

1704 The representation in namespaces other than the Interop namespace should be avoided. However, if

1705 additional StaticFilterCollection instances represent a static filter collection in implementation

1706 namespaces, these StaticFilterCollection instances shall have the same key property values as the one in

1707 the Interop namespace.

1708 If the FilterCollectionCoverageExposure feature (see 7.2.8) is available for a particular

1709 StaticFilterCollection instance, the contained members of the represented static filter collection (see

1710 7.3.17.3), and their containment relationship to the static filter collection are required to be represented in

1711 CIM; see 7.3.12 for the representation of contained static indication filters, see 7.3.17 for the

1712 representation of contained static filter collections, and see 7.3.19 and 7.3.20 for the representation of the

1713 containment relationship.

1714 **7.3.18 IndicationServiceOfFilterCollection: CIM_OwningCollectionElement**

1715 The requirements in this subclause are referencing profile and WBEM server related implementation

1716 requirements.

1717 The IndicationServiceOfFilterCollection adaptation models the relationship between a filter collection and

1718 the indication service that owns the filter collection.

1719 The implementation type of the IndicationServiceOfFilterCollection association adaptation is:

1720 "instantiated".

1721 Table 26 lists the element requirements for the IndicationServiceOfFilterCollection adaptation.

1722 **Table 26 – IndicationServiceOfFilterCollection: Element requirements**

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

1723 Each IndicationService instance (see 7.3.2.4) shall be associated through an
 1724 IndicationServiceOfFilterCollection instance to every StaticFilterCollection instance (see 7.3.17)
 1725 representing a static filter collection managed by the indication service represented by the
 1726 IndicationService instance.

1727 **7.3.19 IndicationFilterInFilterCollection: CIM_MemberOfCollection**

1728 The IndicationFilterInFilterCollection adaptation models the relationship between a filter collection and its
 1729 contained indication filters.

1730 The requirement level of the IndicationFilterInFilterCollection adaptation is conditional.

1731 Condition: The FilterCollectionCoverageExposure feature (see 7.2.8) is implemented.

1732 The implementation type of the IndicationFilterInFilterCollection association adaptation is: "instantiated".

1733 Table 27 lists the element requirements for the IndicationFilterInFilterCollection adaptation.

1734 **Table 27 – IndicationFilterInFilterCollection: Element requirements**

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

1735 Each StaticFilterCollection (see 7.3.17) instance shall be associated through an
 1736 IndicationFilterInFilterCollection instance with each of the IndicationFilter (see 7.3.11) instances
 1737 representing contained indication filters.

1738 **7.3.20 FilterCollectionInFilterCollection: CIM_MemberOfCollection**

1739 The requirements in this subclause are referencing profile and WBEM server related implementation
 1740 requirements.

1741 The FilterCollectionInFilterCollection adaptation models the relationship between a filter collection and its
 1742 contained other filter collections.

1743 The requirement level of the FilterCollectionInFilterCollection adaptation is conditional.

1744 Condition: All of the following:

- 1745 • The static filter collections in the managed environment are capable of containing other static
 1746 filter collections
- 1747 • The FilterCollectionCoverageExposure feature (see 7.2.8) is implemented.

1748 The implementation type of the FilterCollectionInFilterCollection association adaptation is: "instantiated".

1749 Table 28 lists the element requirements for the FilterCollectionInFilterCollection adaptation.

1750 **Table 28 – FilterCollectionInFilterCollection: Element requirements**

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

1751 Each StaticFilterCollection instance (see 7.3.17) representing a static filter collection that contains other
 1752 static filter collections shall be associated through a FilterCollectionInFilterCollection instance with each of
 1753 the StaticFilterCollection instances (see 7.3.17) representing a contained static filter collection.

1754 **7.3.21 ProfileSpecificFilterCollection: CIM_FilterCollection**

1755 **7.3.21.1 General**

1756 The requirements in this subclause are referencing profile and WBEM server related implementation
 1757 requirements.

1758 The ProfileSpecificFilterCollection adaptation models profile-specific filter collections; profile-specific filter
 1759 collections are described in 6.3.3.4.

1760 The requirement level of the ProfileSpecificFilterCollection adaptation is optional.

1761 The ProfileSpecificFilterCollection adaptation should be implemented.

1762 The implementation type of the ProfileSpecificFilterCollection adaptation is: "instantiated".

1763 **7.3.21.2 Element requirements**

1764 **7.3.21.2.1 General**

1765 Table 29 lists the element requirements for the ProfileSpecificFilterCollection adaptation.

1766 **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.
----------------	-----------	-----------------

1767 **7.3.21.2.2 Property: CollectionName**

1768 The value of the CollectionName property shall be formatted as defined by the following ABNF rule:

```
1769 "DMTF:" RegisteredName ":"
1770 "ProfileSpecified" Type "IndicationFilterCollection"
```

1771 OrgID and RegisteredName shall be specified as detailed in 7.3.17.4.2.

1772 Type shall be "Alert" in case the represented profile-specific filter collection covers all alert indications,
 1773 and shall be "Lifecycle" in case the represented profile-specific filter collection covers all lifecycle
 1774 indications defined in the referencing profile identified by RegisteredName.

1775 NOTE This requirement does not preclude more than one instance in the Interop namespace from having
 1776 identical values for the CollectionName property, because, for example, the referencing profile could be
 1777 implemented more than once.

1778 **7.3.21.3 Requirements pertaining to the defined coverage**

1779 Requirements pertaining to the defined coverage are specified on a per instance basis; see 7.3.21.4
 1780 and 7.3.21.4.2.

1781 **7.3.21.4 Instance requirements**

1782 **7.3.21.4.1 Instance requirements for profile-specific filter collections covering all alert indications**
 1783 **specified in a profile**

1784 If and only if a referencing profile defines alert indications, the implementation may expose a
 1785 ProfileSpecificFilterCollection instance in the Interop namespace that covers all alert indications defined
 1786 in that profile. The element requirements defined in 7.3.21.2 apply.

1787 NOTE The existence of that ProfileSpecificFilterCollection instance does not imply that any alert indications are
 1788 actually implemented. Furthermore, in the case where multiple implementations of the referencing profile
 1789 exist in a WBEM server, multiple ProfileSpecificFilterCollection instances may result.

1790 The members of a profile-specific filter collection covering all alert indications defined in a referencing
 1791 profile shall be all indication-specific indication filters covering the alert indications defined in that
 1792 referencing profile; see 7.3.15. This definition in effect defines the defined coverage as all alert indications
 1793 defined in the referencing profile.

1794 NOTE For existing ProfileSpecificFilterCollection instances the instance requirements of association instances
 1795 representing relationships of the represented profile-specific filter collection apply; for example, see 7.3.18,
 1796 7.3.19 or 7.3.20.

1797 **7.3.21.4.2 Instance requirements for profile-specific filter collections covering all lifecycle**
 1798 **indications specified in a profile**

1799 If and only if a referencing profile defines lifecycle indications, the implementation may expose a
 1800 ProfileSpecificFilterCollection instance in the Interop namespace that covers all lifecycle indications
 1801 defined in that profile. The element requirements defined in 7.3.21.2 apply.

1802 NOTE The existence of such a ProfileSpecificFilterCollection instance does not imply that any lifecycle indications
 1803 are actually implemented. Furthermore, in the case where multiple implementations of the referencing
 1804 profile exist in a WBEM server, multiple ProfileSpecificFilterCollection instances may result.

1805 The members of a profile-specific filter collection covering all lifecycle indications defined in a referencing
 1806 profile shall be all indication-specific indication filters covering the lifecycle indications defined in that
 1807 referencing profile or in this profile; see 7.3.15. This definition in effect defines the defined coverage as all
 1808 lifecycle indications defined in the referencing profile.

1809 NOTE For existing ProfileSpecificFilterCollection instances the instance requirements of association instances
 1810 representing relationships of the represented profile-specific filter collection apply; for example, see 7.3.18,
 1811 7.3.19 or 7.3.20.

1812 The requirements specified in this subclause for lifecycle indications defined in referencing profiles shall
 1813 also apply for the lifecycle indications defined in this profile; see 7.3.33 and 7.3.34.

1814 **7.3.22 GlobalFilterCollection: CIM_FilterCollection**

1815 **7.3.22.1 General**

1816 The requirements in this subclause are referencing profile and WBEM server related implementation
 1817 requirements; see 7.1.

1818 The GlobalFilterCollection adaptation models global filter collection; global filter collections are described
 1819 in 6.3.3.5.

1820 The implementation type of the GlobalFilterCollection adaptation is: "instantiated".

1821 **7.3.22.2 Element requirements**

1822 Table 30 lists the element requirements for the ProfileSpecificFilterCollection adaptation.

1823 **Table 30 – GlobalFilterCollection: Element requirements**

Element	Requirement	Description
Base adaptations		
StaticFilterCollection	Mandatory	See 7.3.17.

1824 **7.3.22.3 Requirements pertaining to the defined coverage**

1825 Requirements pertaining to the defined coverage are specified on a per instance basis; see 7.3.22.4.1,
 1826 7.3.22.4.2, 7.3.22.4.3 and 7.3.22.4.4.

1827 **7.3.22.4 Instance requirements**

1828 **7.3.22.4.1 Instance requirements for the global filter collection covering all alert indications**
 1829 **specified in profiles**

1830 If any alert indications specified in referencing profiles or in this profile are implemented, the
 1831 implementation may expose a GlobalFilterCollection instance in the Interop namespace that covers all
 1832 alert indications defined in profiles. In implementations where it is not possible to determine whether alert
 1833 indications specified in referencing profiles are implemented, the instance may be exposed if the delivery
 1834 of alert indications is implemented in general.

1835 In the GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
 1836 following ABNF rule:

1837 "DMTF:Indications:"
 1838 "GlobalProfileSpecifiedAlertIndicationFilterCollection".

1839 In this case the members of the represented global filter collection shall be all profile-specific filter
 1840 collections covering the alert indications defined in any implemented referencing profile or in this profile;
 1841 see 7.3.21.4. This definition in effect specifies the defined coverage as all alert indications defined in
 1842 referencing profiles and in this profile; if instantiated by an implementation, the coverage would be all
 1843 implemented alert indications out of that set.

1844 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
 1845 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
 1846 or 7.3.20.

1847 **7.3.22.4.2 Instance requirements for the global filter collection covering all lifecycle indications** 1848 **specified in profiles**

1849 If any lifecycle indications specified in referencing profiles or in this profile are implemented, the
 1850 implementation may expose a GlobalFilterCollection instance in the Interop namespace that covers all
 1851 lifecycle indications defined in profiles. In implementations where it is not possible to determine whether
 1852 lifecycle indications specified in referencing profiles are implemented, the instance may be exposed if the
 1853 delivery of lifecycle indications is implemented in general.

1854 In GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
 1855 following ABNF rule:

```
1856     "DMTF:Indications:"  
1857     "GlobalProfileSpecifiedLifecycleIndicationFilterCollection".
```

1858 The members of the represented global filter collection shall be all profile-specific filter collections
 1859 covering the lifecycle indications defined in any implemented referencing profile or in this profile; see
 1860 7.3.21.4.2. This definition in effect specifies the defined coverage as all lifecycle indications defined in
 1861 referencing profiles and in this profile; if instantiated by an implementation, the coverage would be all
 1862 implemented lifecycle indications out of that set.

1863 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
 1864 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
 1865 or 7.3.20.

1866 **7.3.22.4.3 Instance requirements for the global filter collection covering all indications specified** 1867 **in profiles**

1868 If any indications specified in referencing profiles or in this profile are implemented, the implementation
 1869 may expose a GlobalFilterCollection instance in the Interop namespace that covers all indications defined
 1870 in profiles. In implementations where it is not possible to determine whether indications specified in
 1871 referencing profiles are implemented, the instance may be exposed if the delivery of indications is
 1872 implemented in general.

1873 In the GlobalFilterCollection instance, the value of the CollectionName property shall be as defined by the
 1874 following ABNF rule:

```
1875     "DMTF:Indications:"  
1876     "GlobalProfileSpecifiedIndicationFilterCollection"
```

1877 The members of the represented global filter collection shall be the following global filter collections (if
 1878 existing):

- 1879 • the global filter collection covering all alert indications defined in any implemented referencing
 1880 profile, as required in 7.3.22.4.1
- 1881 • the global filter collection covering all lifecycle indications defined in any implemented
 1882 referencing profile, as required in 7.3.22.4.2

1883 This definition in effect specifies the defined coverage as all indications defined in referencing profiles and
1884 in this profile; if instantiated by an implementation, the coverage would be all implemented indications out
1885 of that set.

1886 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
1887 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
1888 or 7.3.20.

1889 **7.3.22.4.4 Instance requirements for the global filter collection covering all lifecycle indications**

1890 If the implementation supports the delivery of lifecycle indications, the implementation shall expose a
1891 GlobalFilterCollection instance in the Interop namespace that covers all lifecycle indications defined in
1892 profiles.

1893 In GlobalFilterCollection instance the value of the CollectionName property shall be as defined by the
1894 following ABNF rule:

1895 `"DMTF:Indications:GlobalLifecycleIndicationFilterCollection".`

1896 The members of the represented global filter collection shall be all profile-specific filter collections
1897 covering the global indication filters that each cover all indications of one of the three subtypes of lifecycle
1898 indications (CIM_InstCreation, CIM_InstDeletion and CIM_InstModification); see 7.3.16.3.2.

1899 This definition in effect specifies the defined coverage as all lifecycle indications defined in referencing
1900 profiles and in this profile.

1901 NOTE For existing GlobalFilterCollection instances the instance requirements of association instances
1902 representing relationships of the represented global filter collection apply; for example, see 7.3.18, 7.3.19
1903 or 7.3.20.

1904 **7.3.23 ListenerDestination: CIM_ListenerDestination**

1905 **7.3.23.1 General**

1906 The ListenerDestination adaptation models listener destinations; listener destinations are described in
1907 6.4.5.

1908 The implementation type of the ListenerDestination adaptation is: "instantiated".

1909 **7.3.23.2 Semantical requirements**

1910 For a particular listener destination, an implementation shall deliver any indication that passed the
1911 indication gate (see 6.2 or 6.3) referenced by any subscription (see 6.4.1) that also references the listener
1912 destination, to the listener referenced by that listener destination. See also the semantical requirements
1913 on indication filters defined in 7.3.11.2, on filter collections defined in 7.3.17.2, and on subscriptions
1914 defined in 7.3.25.2.

1915 NOTE It is possible that a particular indication is delivered more than once to a particular listener for various
1916 reasons, such as that the listener is referenced by more than one listener destination, or that the indication
1917 is within the coverage of more than one indication gate, each of which is referenced by a subscription
1918 referencing the listener destination referencing the listener.

1919 **7.3.23.3 Element requirements**

1920 **7.3.23.3.1 General**

1921 Table 31 lists the element requirements of the ListenerDestination adaptation.

1922

Table 31 – ListenerDestination Element requirements

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

1923 **7.3.23.3.2 Property: Destination**

1924 The value of the Destination property shall identify the listener referenced by the listener destination.

1925 A value of Null for the Destination property indicates a free listener destination (see 6.4.5).

1926 If the value of the Destination property is not Null, it shall be a valid IETF Uniform Resource Identifier
 1927 value (as defined in [RFC3986](#)) including the scheme, host and port as part of the URI Location.

1928 **7.3.23.3.3 Property: PersistenceType**

1929 The value of the PersistenceType property shall describe the durability of the represented listener
 1930 destination.

1931 The property values shall be constrained to 3 (Transient), 2 (Permanent), and Null.

1932 If the listener destination is permanent, then the value of the PersistenceType property shall be either Null
 1933 or 2 (Permanent). Permanent listener destinations are long-lived and are expected to be available for
 1934 indication delivery. For example, a typical listener referenced by a permanent listener destination would
 1935 be a system log file. The inability of an implementation to deliver indications to a listener referenced by a
 1936 permanent listener destination will be treated as an error condition by the implementation, as defined in
 1937 7.4.3.5.

1938 If the listener destination is transient, then the value of the PersistenceType property shall be 3
 1939 (Transient). Transient listener destinations are short-lived and have less strong requirements (than
 1940 permanent listener destinations) regarding their availability for indication delivery. For example, a typical
 1941 listener referenced by a transient listener destination would be a task progress meter in a graphical

1942 management application. The inability of an implementation to deliver indications to a listener described
 1943 by a transient listener destination will be handled by removing the listener destination and its
 1944 subscriptions from the implementation, as defined in 7.4.3.6.

1945 **7.3.23.3.4 Operation: CreateInstance()**

1946 Table 32 lists the error reporting requirements for the CreateInstance() operation on ListenerDestination
 1947 instances. If any of the error situations described in the Description column of Table 32 matches, the
 1948 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1949 reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

1950 **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.

1951 If the CreateInstance() operation is successful, the requested listener destination shall be created, and —
 1952 as a consequence — shall be represented by a ListenerDestination instance in the requested
 1953 namespace. In addition, if the requested namespace is not the Interop namespace, the implementation
 1954 shall expose an additional ListenerDestination instance representing the listener destination in the Interop
 1955 namespace (see 7.3.23.4).

1956 If the CreateInstance() operation fails, no listener destination shall be created, and — as a consequence
 1957 — no representing ListenerDestination instances shall be exposed in any namespace.

1958 The implementation may ignore the values of key properties in the embedded CIM_ListenerDestination
 1959 instance passed as the value of the NewInstance parameter.

1960 Clients should abstain from providing the values of key properties in the embedded
 1961 CIM_ListenerDestination instance passed as the value of the NewInstance parameter.

1962 Clients should abstain from requesting the creation of ListenerDestination instances in namespaces other
 1963 than the Interop namespace.

1964 Clients should favor the re-use of an existing listener destination referencing a particular listener over the
 1965 creation of a new listener destination referencing the same listener.

1966 **7.3.23.3.5 Operation: DeleteInstance()**

1967 Table 33 lists the error reporting requirements for the DeleteInstance() operation on ListenerDestination
 1968 instances, and related CIM status codes. If any of the error situations described in the Description column

1969 of Table 33 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 1970 addition, the error reporting requirements defined in [DSP0223](#) for the DeleteInstance() operation apply.

1971 **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).

1972 If the DeleteInstance() operation is successful, the represented listener destination shall be deleted and
 1973 — as a consequence — shall no longer be represented by ListenerDestination instances in any
 1974 namespace exposed by the implementation.

1975 NOTE The instance requirements of associations representing relationships of the deleted listener destination
 1976 imply that respective association instances in any namespace exposed by the implementation cease to
 1977 exist; in this case this applies to IndicationServiceOfListenerDestination instances (see 7.3.24). However,
 1978 note that the DeleteInstance() operation for the listener destination is required to fail if subscriptions exist.

1979 If the DeleteInstance() operations fails, the listener destination shall not be deleted, and — as a
 1980 consequence — any representing ListenerDestination instances shall continue to exist as before.

1981 **7.3.23.3.6 Operation: ModifyInstance()**

1982 The ModifyInstance operation may be available for an instance of CIM_ListenerDestination.

1983 The implementation of the ModifyInstance() operation enables clients to modify existing listener
 1984 destinations.

1985 The requirement level of the ModifyInstance() operation is optional.

1986 Table 34 lists the error reporting requirements for the ModifyInstance() operation on ListenerDestination
 1987 instances. If any of the error situations described in the Description column of Table 34 matches, the
 1988 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 1989 reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation apply.

1990 **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).

1991 If the ModifyInstance() operation is successful, the requested modification on the listener destination
 1992 shall be applied, and — as a consequence — shall be reflected in all ListenerDestination instances that
 1993 represent the modified listener destination and are exposed by the implementation.

1994 If the ModifyInstance() operation fails, the requested modification on the listener destination shall not be
 1995 applied, and — as a consequence — all ListenerDestination instances that represent the listener
 1996 destination shall remain unchanged.

1997 **7.3.23.4 Instance requirements**

1998 Listener destinations (see 6.4.5) shall be represented by ListenerDestination instances in the Interop
 1999 namespace.

2000 The representation in namespaces other than the Interop namespace should be avoided. However, if
 2001 additional ListenerDestination instances represent the listener destination in implementation namespaces,
 2002 these ListenerDestination instances shall have the same key property values as the one in the Interop
 2003 namespace.

2004 **7.3.24 IndicationServiceOfListenerDestination: CIM_ServiceAffectsElement**

2005 The IndicationServiceOfListenerDestination adaptation models the relationship between indication
 2006 services and the listener destinations they manage. Indication services are described in 6.5.2; listener
 2007 destinations are described in 6.4.5.

2008 The implementation type of the IndicationServiceOfListenerDestination association adaptation is:
 2009 "instantiated".

2010 Table 35 lists the elements requirements of the IndicationServiceOfListenerDestination adaptation.

2011 **Table 35 – IndicationServiceOfListenerDestination: Element requirements**

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

2012 Each IndicationService (see 7.3.2) instance shall be associated through an
 2013 IndicationServiceOfListenerDestination instance with each ListenerDestination (see 7.3.23) instance
 2014 representing a listener destination managed by the indication service represented by the
 2015 IndicationService instance.

2016 **7.3.25 AbstractSubscription: CIM_AbstractIndicationSubscription**

2017 **7.3.25.1 General**

2018 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2019 The AbstractSubscription adaptation models subscriptions for the delivery of indications from an
 2020 indication gate to a listener referenced by a listener destination; subscriptions are described in 6.4.

2021 The implementation type of the AbstractSubscription association adaptation is: "abstract".

2022 **7.3.25.2 Semantical requirements**

2023 An implementation shall deliver any indication that passed the indication gate referenced by the
 2024 subscription (that is, any indication generated by the implementation that is within the coverage of the
 2025 indication gate) to the listener referenced by the listener destination referenced by the subscription.

2026 A listener that is referenced by the listener destination referenced by a subscription needs to be prepared
 2027 to receive any indication that is within the coverage of the indication gate referenced by that subscription.
 2028 Of course, listeners may ignore received indications.

2029 **7.3.25.3 Element requirements**

2030 Table 36 lists the element requirements for the AbstractSubscription adaptation.

2031 **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.
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.		

2032 **7.3.25.3.1 Property: OnFatalErrorPolicy**

2033 The value of the OnFatalErrorPolicy property shall indicate the behavior that the implementation exposes
2034 with respect to represented subscriptions in case of failures that imply that some aspect of indication
2035 generation processing or indication delivery is no longer functioning and indications may be lost.

2036 A value of 4 (Remove) shall indicate that the implementation performs implicit subscription removal as
2037 detailed in 7.4.3.6; this shall be the default behavior.

2038 **7.3.25.3.2 Property: RepeatNotificationPolicy**

2039 The value of the RepeatNotificationPolicy property shall indicate the policy that the implementation
2040 applies with respect to the avoidance of repeated indication delivery of repeated indications as described
2041 in 6.1.6.

2042 Table 37 lists constraints for the value of the RepeatNotificationPolicy property.

2043 **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)

2044 **7.3.25.3.3 Property: RepeatNotificationInterval**

2045 The requirement level of the RepeatNotificationInterval property is conditional exclusive.

2046 Condition: Either the SuppressRepeatNotificationPolicy feature (see 7.2.5) or the
2047 DelayRepeatNotificationPolicy feature (see 7.2.6) is available.

2048 If the implementation applies the SuppressRepeatNotificationPolicy feature (see 7.2.5) for the
2049 represented subscription, as indicated by the value 3 (Suppress) for the RepeatNotification property, the
2050 value of the RepeatNotificationInterval property shall be the length of the time interval in seconds that the

2051 implementation waits after initial delivery of a number of repeated indications as indicated by the value of
2052 the RepeatNotificationCount property before delivering the next repeated indication.

2053 If the implementation applies the DelayRepeatNotificationPolicy feature (see 7.2.6) for the represented
2054 subscription, as indicated by the value 4 (Delay) for the RepeatNotification property, the value of the
2055 RepeatNotificationInterval property shall be the length of the monitoring time interval in seconds during
2056 which the implementation monitors the indication gate referenced by the subscription for a number of
2057 additional repeated indications. Furthermore, only if during that monitoring interval at least the number of
2058 repeated indications as indicated by the value of the RepeatNotificationCount accrue, delivers only the
2059 first indication as a substitute for all the repeated indications accrued during the monitoring time interval.

2060 **7.3.25.3.4 Property: RepeatNotificationGap**

2061 The requirement level of the RepeatNotificationGap property is conditional exclusive.

2062 Condition: The DelayRepeatNotificationPolicy feature (see 7.2.6) is implemented.

2063 The value of the RepeatNotificationGap property shall be the length of the delay time interval in seconds
2064 that the implementation waits after delivering the first of a number of repeated indications that accrued
2065 during the monitoring time interval, before starting another monitoring time interval, as described in
2066 7.3.25.3.5 with respect to implementations of the DelayRepeatNotificationPolicy feature.

2067 **7.3.25.3.5 Property: RepeatNotificationCount**

2068 The requirement level of the RepeatNotificationCount property is conditional exclusive.

2069 Condition: Either the SuppressRepeatNotificationPolicy feature (see 7.2.5) or the
2070 DelayRepeatNotificationPolicy feature (see 7.2.6) is implemented.

2071 If the implementation applies the SuppressRepeatNotificationPolicy feature (see 7.2.5) for the
2072 represented subscription, as indicated by the value 3 (Suppress) for the RepeatNotification property, the
2073 value of the RepeatNotificationCount property shall be the number of repeated indications that the
2074 implementation delivers before suppressing the delivery of further repeated indications within the time
2075 interval exposed by the value of the RepeatNotificationInterval property.

2076 If the implementation applies the DelayRepeatNotificationPolicy feature (see 7.2.6) for the represented
2077 subscription, as indicated by the value 4 (Delay) for the RepeatNotification property, the value of the
2078 RepeatNotificationCount property shall be the number of repeated indications that the implementation is
2079 required to monitor and delay during the monitoring time interval exposed by the value of the
2080 RepeatNotificationInterval property. Only if during that monitoring time interval the number of accrued
2081 repeated indications reaches that number, the implementation shall deliver the first of repeated indication
2082 as a substitute for the accrued repeated indications. In other words, the quotient of the values of the
2083 RepeatNotificationCount and the RepeatNotificationInterval properties expresses a rate of repeated
2084 indications that must have been reached or exceeded during the monitoring time interval before one
2085 indication is delivered at the end of the monitoring time interval.

2086 **7.3.25.3.6 Operation: DeleteInstance()**

2087 The error situations and CIM status codes defined in [DSP0223](#) for the DeleteInstance() operation apply.

2088 If the DeleteInstance() operation succeeds, the represented subscription shall be deleted and — as a
2089 consequence — shall no longer be represented by any AbstractSubscription instances in any namespace
2090 exposed by the implementation.

2091 If the DeleteInstance() operation fails, the subscription shall not be deleted, and — as a consequence —
2092 any representing AbstractSubscription instances shall continue to exist as before.

2093 **7.3.25.3.7 Operation: ModifyInstance()**

2094 The requirement level of the ModifyInstance() operation is optional.

2095 Table 38 lists the error reporting requirements for the ModifyInstance() operation on AbstractSubscription
 2096 instances, and related CIM status codes. If any of the error situations described in the Description column
 2097 of Table 38 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 2098 addition, the error reporting requirements defined in [DSP0223](#) for the ModifyInstance() operation are
 2099 applicable.

2100 **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.

2101 If the ModifyInstance() operation is successful, the requested modification on the represented
 2102 subscription shall be applied, and — as a consequence — shall be reflected in all AbstractSubscription
 2103 instances that represent the modified subscription.

2104 If the ModifyInstance() operation fails, the requested modification on the subscription shall not be
 2105 applied, and — as a consequence — all AbstractSubscription instances that represent the subscription
 2106 shall remain unchanged.

2107 **7.3.25.4 Instance requirements**

2108 Subscriptions (see 6.4.1) shall be represented by AbstractSubscription instances in the Interop
 2109 namespace that relate either IndicationFilter instances (see 7.3.11) or StaticFilterCollection instances
 2110 (see 7.3.17) with ListenerDestination instances (see 7.3.23).

2111 The representation in namespaces other than the Interop namespace should be avoided. However, if
 2112 both the indication filter/filter collection and the related listener destination represented by the referenced
 2113 instances in the Interop namespace are also represented by additional instances in other namespaces,
 2114 respective AbstractSubscription instances shall represent the subscription in these other namespaces as
 2115 well.

2116 **7.3.26 FilterSubscription: CIM_IndicationSubscription**

2117 **7.3.26.1 General**

2118 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2119 The FilterSubscription adaptation models subscriptions for the delivery of indications from an indication
 2120 filter to a listener referenced by a listener destination; subscriptions are described in 6.4.

2121 The requirement level of the FilterSubscription adaptation is conditional.

2122 Condition: The IndividualFilterSubscription feature (see 7.2.7) is implemented.

2123 The implementation type of the FilterSubscription association adaptation is: "instantiated".

2124 **7.3.26.2 Semantical requirements**

2125 The semantical requirements of 7.3.25.2 apply respectively for the FilterSubscription adaptation.

2126 **7.3.26.3 Element requirements**

2127 **7.3.26.3.1 General**

2128 Table 39 lists the element requirements for the FilterSubscription adaptation.

2129 **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 .

2130 **7.3.26.3.2 Operation: CreateInstance()**

2131 Table 40 lists the error reporting requirements for the CreateInstance() operation on FilterSubscription
 2132 instances. If any of the error situations described in the Description column of Table 40 matches, the
 2133 operation shall fail and the corresponding CIM status code shall be returned. In addition, the error
 2134 reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

2135 **Table 40 – CreateInstance(): Error reporting requirements**

Reporting mechanism	Requirement level	Description
---------------------	-------------------	-------------

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

2136 If the CreateInstance() operation is successful, the requested filter subscription was created, and
 2137 consequently — as required by 7.3.26.4 — shall be represented by a FilterSubscription instance in the
 2138 requested namespace. In addition, if the requested namespace is not the Interop namespace, the
 2139 implementation shall expose an additional FilterSubscription instance representing the subscription in the
 2140 Interop namespace (see 7.3.26.4).

2141 If the CreateInstance() operation fails, no subscription shall be created, and — as a consequence — no
 2142 representing FilterSubscription instances shall be exposed in any namespace.

2143 Clients should abstain from requesting the creation of FilterSubscription instances in namespaces other
 2144 than the Interop namespace.

2145 **7.3.26.4 Instance requirements**

2146 The requirements of 7.3.25.4 apply respectively for FilterSubscription instances.

2147 **7.3.27 CollectionSubscription: CIM_FilterCollectionSubscription**

2148 **7.3.27.1 General**

2149 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2150 The CollectionSubscription adaptation models subscriptions for the delivery of indications from a filter
 2151 collection to a listener referenced by a listener destination; subscriptions are described in 6.4.

2152 The implementation type of the FilterCollectionSubscription association adaptation is: "instantiated".

2153 **7.3.27.2 Semantical requirements**

2154 The semantical requirements of 7.3.25.2 apply respectively for the CollectionSubscription adaptation.

2155 **7.3.27.3 Element requirements**

2156 **7.3.27.3.1 General**

2157 Table 41 lists the element requirements for the CollectionSubscription adaptation.

2158 **Table 41 – CollectionSubscription: Element requirements**

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

2159 **7.3.27.3.2 Operation: CreateInstance()**

2160 Table 42 lists the error reporting requirements for the CreateInstance() operation on
 2161 CollectionSubscription instances. If any of the error situations described in the Description column of
 2162 Table 42 matches, the operation shall fail and the corresponding CIM status code shall be returned. In
 2163 addition, the error reporting requirements defined in [DSP0223](#) for the CreateInstance() operation apply.

2164 **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.		

2165 If the CreateInstance() operations is successful, the requested filter subscription was created, and
 2166 consequently — as required by 7.3.27.4 — shall be represented by a CollectionSubscription instance in
 2167 the requested namespace. In addition, if the requested namespace is not the Interop namespace, the
 2168 implementation shall expose an additional CollectionSubscription instance representing the subscription
 2169 in the Interop namespace (see 7.3.27.4).

2170 If the CreateInstance() operation fails, no subscription shall be created, and — as a consequence — no
 2171 representing CollectionSubscription instances shall be exposed in any namespace.

2172 Clients should abstain from requesting the creation of CollectionSubscription instances in namespaces
 2173 other than the Interop namespace.

2174 **7.3.27.4 Instance requirements**

2175 The instance requirements of 7.3.25.4 apply respectively for CollectionSubscription instances.

2176 **DEPRECATED**

2177 **7.3.28 ProfileOfFilterCollection: CIM_ConcreteDependency**

2178 The ProfileOfFilterCollection adaptation models the relationship between a filter collection defined in a
 2179 referencing profile and the profile registration of that referencing profile.

2180 The implementation type of the ProfileOfFilterCollection association adaptation is: "instantiated".

2181 Each StaticFilterCollection instance (see 7.3.17) representing a filter collection defined in a referencing
 2182 profile shall be associated through a ProfileOfFilterCollection instance with the ProfileRegistration
 2183 instance (see [DSP1033](#)) representing the implemented version of the referencing profile.

2184 NOTE This profile assumes that a future version of the Profile Registration profile (see [DSP1033](#)) will be based
 2185 on version 1.1 of the Profile Usage Guide (see [DSP1001](#)), and define the ProfileRegistration adaptation;
 2186 until then, substitute that by the definition of the CIM_RegisteredProfile "profile class" defined in version
 2187 1.0 of [DSP1033](#).

2188 Table 43 lists the element requirements for the ProfileOfFilterCollection adaptation.

2189 **Table 43 – ProfileOfFilterCollection: Element requirements**

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

2190 **DEPRECATED**

2191 **7.3.29 BasicIndication: CIM_Indication**

2192 **7.3.29.1 General**

2193 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2194 The BasicIndication adaptation models indications; indications are described in 6.1.

- 2195 The implementation type of the BasicIndication indication adaptation is: "abstract".
- 2196 **7.3.29.2 Event definition requirements**
- 2197 Referencing profiles that model indications through adaptations based on the BasicIndication adaptation
 2198 shall define event that the indication is designed to report. This event definition shall be accomplished by
 2199 means of an event definition query statement stated in CQL (see [DSP0202](#)).
- 2200 The purpose of an event definition query statement is to formally define the event(s) that an indication
 2201 adaptation is designed to report, such that by inspecting the event definition query statements an
 2202 implementer knows how to implement the indication adaptation. A CIM representation of event definition
 2203 query statements is not defined, thus there is no requirement for implementations or clients to be able to
 2204 programmatically interpret event definition query statements.
- 2205 **NOTE** Event definition query statements are different from indication filter query statements. An indication filter
 2206 query statement (see 7.3.11.3.5) defines the coverage of an indication filter, and is exposed to clients by
 2207 the value of the Query property in the IndicationFilter instance representing the indication filter. The
 2208 IndicationSpecificIndicationFilter adaptation (see 7.3.15) models indication-specific indication filters (see
 2209 6.2.4) and addresses the needs of clients requiring notifications about events reported by particular
 2210 indications specified in a profile.
- 2211 The CQL query statement defining the event shall comply with the following ABNF rule:
- 2212 `"SELECT" WS PropertySet WS "FROM" WS IndicationClass WS "WHERE" WS`
 2213 `SelectionExpression`
- 2214 `PropertySet` shall be `"*"`, or a comma-separated list of property names.
- 2215 `IndicationClass` shall be the adapted indication class, that is, `CIM_Indication` or a subclass thereof.
- 2216 `SelectionExpression` shall be a constant string that defines a selection expression conformant with
 2217 the rules for selection expressions defined by [DSP0202](#).
- 2218 `WS` represents one or more whitespace characters.
- 2219 The requirements in this subclause may be refined by requirements defined in adaptations based on the
 2220 BasicIndication adaptation, including the case that a refined query statement references an external
 2221 element (such as an alert message definition in a message registry) that defines the event.
- 2222 **7.3.29.3 Indication origin**
- 2223 Each indication shall be assigned an origin namespace (see 6.1.2.4).
- 2224 In general, an implementation is free to select any local namespace as the origin namespace for a
 2225 generated indication; however, adaptations based on the BasicIndication adaptation such as the
 2226 AlertIndication adaptation (see 7.3.31) and the LifecycleIndication (see 7.3.32) establish additional
 2227 constraints.
- 2228 The indication origin is not represented in the CIM representation of an indication as defined by the
 2229 `CIM_Indication` class.
- 2230 The implementation class of the indication is required to reside in the origin namespace.
- 2231 **NOTE** As with any implementation class, the existence of an indication implementation class within a namespace
 2232 is does not sufficiently indicate that the indication is really implemented. Additional requirements — such
 2233 as the presence and integration of functional code implementing the indication — apply, but are outside of
 2234 the scope of this profile.
- 2235 The indication origin is required to be considered during indication filtering; see 6.1.4 and 7.3.11.2.

2236 **7.3.29.4 Element requirements**

2237 **7.3.29.4.1 General**

2238 Table 44 lists the element requirements for the BasicIndication adaptation.

2239 **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.

2240 **7.3.29.4.2 Property: IndicationFilterName**

2241 The value of the IndicationFilterName property shall contain the name of the indication gate that the
 2242 indication passed before being delivered to the listeners subscribed to that indication gate. For indication
 2243 filters, the name is exposed by the value of the Name property in representing IndicationFilter instances
 2244 (see 7.3.11). For filter collections, the name is exposed by the value of the CollectionName property in
 2245 representing StaticFilterCollection instances (see 7.3.17).

2246 Because an indication is generated independently and before it is subjected to filtering, the name of the
 2247 filtering indication gate is not known at indication-generation time. Instead, a generated indication might
 2248 match a large number of indication gates. During indication filtering (see 6.1.4 and 7.3.11.2), each time a
 2249 generated indication matches an indication gate with existing subscriptions, and before delivering that
 2250 indication to subscribed listeners, the implementation shall set the value of the IndicationFilterName
 2251 property in the BasicIndication instance representing the indication to the identification of that indication
 2252 gate, as follows:

- 2253 • in case of indication filters, the identification shall be the value of the Name property of the
 2254 IndicationFilter instance representing the indication filter
- 2255 • in case of filter collections, the identification shall be the value of the CollectionName property of
 2256 the StaticFilterCollection instance representing the filter collection.

2257 NOTE 1 The requirement for referencing filter collections was added with version 1.2. of this profile.

2258 NOTE 2 A listener may use the value of the IndicationFilterName property to determine which indication gate was
 2259 passed by the indication before being delivered to the listener.

2260 **7.3.29.5 Indication generation requirements**

2261 Adaptations based on the BasicIndication adaptation are required to define the event that the modeled
 2262 indication is designed to report; see 7.3.29.2.

2263 If the event defined by such an adaptation occurs, and if subscriptions exist for any indication gate
 2264 covering the modeled indication, an instance of the indication adaptation based on the BasicIndication
 2265 shall be generated.

2266 NOTE The way this requirement is stated it provides for the optimized approach of checking for the presence of
 2267 matching indication gate with subscriptions already at indication generation time; however, even in this
 2268 case indication filtering is required as a subsequent step (see 6.1.4) in order to ensure that all matching
 2269 indication gates are considered, and indication delivery occurs to all listeners subscribed to any of the
 2270 indication gates covering the indication.

2271 **7.3.30 ReliableIndication: CIM_Indication**

2272 **7.3.30.1 General**

2273 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2274 The ReliableIndication adaptation models reliable indications; the concept of reliable indications is
 2275 introduced in 6.1.5. Additional requirements for reliable indication delivery are specified in 7.4.

2276 The implementation type of the ReliableIndication indication adaptation is: "abstract".

2277 NOTE The ReliableIndications adaptation is intentionally not based on the BasicIndication adaptation, such that it
 2278 can be implemented independently as a separate option. Reliable indication delivery is typically
 2279 implemented centrally once for the delivery of all indications implemented by an implementation.

2280 **7.3.30.2 Element requirements**

2281 **7.3.30.2.1 General**

2282 Table 45 lists the element requirements for the ReliableIndication adaptation.

2283 **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.

2284 **7.3.30.2.2 Property: SequenceContext**

2285 The value of the SequenceContext property shall contain the sequence context portion of the sequence
 2286 identifier (see 3.30 and 7.4.2). See the CIM schema description for additional constraints and the required
 2287 semantics, and see 7.4 for additional requirements on reliable indication delivery.

2288 NOTE 1 The CIM schema definition of the CIM_Indication class requires for the SequenceContext property that the
 2289 implementation maintains the context for this property separately for each registered listener destination,
 2290 and that restarts of the WBEM server cause the value to change. This requirement enables a listener to
 2291 detect WBEM server restarts, and to differentiate the indication streams from a particular WBEM server
 2292 that were processed (within that WBEM server) through different listener destinations referring to the
 2293 listener.

2294 NOTE 2 Indications can be lost when a listener fails and restarts, with the WBEM server continuing to send
 2295 indications while the listener is inactive. In that case, upon restart of the listener, if does not persist the last
 2296 received sequence identifier, the listener would establish the sequence identifier of the first received
 2297 indication after the restart as check value, failing to notice that while it was inactive additional indications
 2298 were sent (and lost). One approach for discovering an actual loss of indications might be to persist the
 2299 latest sequence identifier as part of a listener termination routine, and upon restart use the persisted value
 2300 as a check value (instead of that taken from the first arriving indication after the restart).

2301 **7.3.30.2.3 Property: SequenceNumber**

2302 The value of the SequenceNumber property shall contain the sequence number portion of the sequence
 2303 identifier (see 3.30 and 7.4.2). See the CIM schema description for additional constraints and the required
 2304 semantics, and see 7.4 for additional requirements on reliable indication delivery.

2305 NOTE The CIM schema definition of CIM_Indication class requires for the SequenceNumber property in the
 2306 stream of instances processed through a particular listener destination, that the value starts at 0 whenever
 2307 the value of the SequenceContext property changes.

2308 **7.3.31 AlertIndication: CIM_AlertIndication**2309 **7.3.31.1 General**

2310 The AlertIndication adaptation models alert indications; alert indications are described in 6.1.3.

2311 The implementation type of the AlertIndication indication adaptation is: "abstract".

2312 It is expected that the AlertIndication adaptation is used as a base adaptation for modeling alert
2313 indications in referencing profiles.

2314 **7.3.31.2 Event definition requirements**

2315 This subclause refines the event definition requirements established by the BasicIndication adaptation;
2316 see 7.3.29.2.

2317 The query statement defined by the following ABNF rules define the event(s) that are reported by
2318 AlertIndication instances:

- 2319 • If the AlertIndication adaptation identifies only one related alert message (see 7.3.31.3), the
2320 event query statement is defined as follows:

```
2321 EventQuerySingle = "SELECT" WS PropertySet WS "FROM" WS
2322 AlertIndicationClass WS "WHERE" WS "OwningEntity='" OwningEntity "'"
2323 WS "AND" WS "MessageID=" MessageId WS AdditionalWhereElements
```

- 2324 • If the AlertIndication adaptation identifies more than one related alert message (see 7.3.31.3),
2325 the event query statement is defined as follows:

```
2326 EventQueryMulti = "SELECT" WS PropertySet WS "FROM" WS
2327 AlertIndicationClass WS "WHERE" WS "OwningEntity='" OwningEntity "'"
2328 WS "AND" WS "MessageID LIKE" WS "'" MessageSet "'" [ WS
2329 AdditionalSelectionExpression ]
```

```
2330 MessageSet = MessageIdentification [ "|" MessageSet ]
```

2331 NOTE Recall that the purpose of the event definition query statement is to formally define the event(s) that an
2332 indication is designed to report; see 7.3.29.2. Event definition query statements are not represented in
2333 CIM; thus there is no requirement for implementations or clients to interpret event definition query
2334 statements.

2335 PropertySet shall be "*", or a comma-separated list of property names.

2336 AlertIndicationClass shall be CIM_AlertIndication, or, if adaptations based on the
2337 AlertIndication adaptation adapt a class derived from CIM_AlertIndication, shall be replaced by the name
2338 of the adapted alert indication class.

2339 OwningEntity shall be the name of the organization defining the alert indication. In profiles owned by
2340 DMTF, the value shall be "DMTF".

2341 MessageIdentification shall identify each referenced alert message, as required by 7.3.31.3.

2342 Referencing profiles in their adaptations based on the AlertIndication adaptation may refine the event
2343 definition; however, such refinements shall remain within the constraints established by the query
2344 statement specified in this subclause.

2345 If a referencing profile defining an adaptation based on the AlertIndication adaptation does not require
2346 refining the query statement specified in this subclause, then a repetition of the query statement is not
2347 required as part of the adaptation in the referencing profile, and compliance with this subclause is
2348 achieved through designating a related alert message as required in 7.3.31.3.

2349 `AdditionalSelectionExpression` shall be a constant string that defines a selection expression
2350 conformant with the rules for selection expressions defined by [DSP0202](#). For example, the value of the
2351 `PerceivedSeverity` property could be constrained to specific values.

2352 7.3.31.3 Related alert messages

2353 Referencing profiles defining adaptations based on the `AlertIndication` adaptation as part of their alert
2354 indication adaptation shall reference one or more related CIM alert message(s) that are defined in a
2355 message registry conformant to [DSP0228](#).

2356 The formal requirements for referencing alert messages through message identifications as part of
2357 adaptation definitions are detailed in [DSP1001](#); as defined there, the main elements of a message
2358 identification are the name of the registry reference referring to the registry defining the alert message,
2359 and the message id as the concatenation of the value of the `PREFIX` attribute and the
2360 `SEQUENCE_NUMBER` attribute from the `MESSAGE_ID` element that defines the message within the
2361 message registry.

2362 CIM alert messages provide for a formalized and widely self-contained approach to define alert
2363 indications. CIM alert messages are defined in message registries. A message registry is an XML
2364 document that contains message definitions. [DSP0228](#) defines an XML schema for message registries.
2365 The schema defines the XML elements that can be used for message definitions. Each element is
2366 formally defined using the XML schema language. Each of these element definitions is annotated with
2367 documentation that may define formal requirements for the use of the message element.

2368 Each message definition in a message registry consists of a standard message identifier and a
2369 description of static and dynamic message elements and of other message components; for details, see
2370 [DSP0228](#).

2371 The `MESSAGE_ID` element within the message definition identifies the message within the scope of the
2372 message registry through a prefix and a sequence number.

2373 The `MESSAGE_DESCRIPTION` element within an alert message definition contains a plain text description
2374 of the event that is reported by the defined alert message. A profile modeling an alert indication shall rely
2375 on the event definition provided in the alert message description. In case the alert-message-based
2376 definition of the event is insufficient in the context of the profile, the profile may augment the event
2377 definition within its definition of the alert indication; however, the amendments to the event definition
2378 stated in a profile shall remain within the constraints defined by the event definition in the alert message
2379 definition in the message repository.

2380 The `<MESSAGE_COMPONENTS>` element within an alert message definition defines a sequence of static
2381 and dynamic elements that together compose the message. The static elements define constant text
2382 parts of the message. The dynamic elements reference property values in identified CIM instances, such
2383 that the property values become dynamic parts of the alert message.

2384 7.3.31.4 Indication origin

2385 If the alert indication is related to a managed object, and the CIM representation of that managed object is
2386 referenced by the value of the `AlertingManagedElement` property in the CIM representation of the alert
2387 indication, then the indication origin as required by 7.3.29.3 should be the namespace in which the CIM
2388 representation of that managed object exists.

2389 7.3.31.5 Element requirements

2390 7.3.31.5.1 General

2391 Table 46 lists the element requirements for the `AlertIndication` adaptation.

2392

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.

2393 7.3.31.5.2 Property: AlertingManagedElement

2394 If the managed element for which the alert indication is reported is represented by one or more CIM
 2395 instances within the implementation, then the value of the AlertingManagedElement property shall identify
 2396 the most prominent of these CIM instances, using the format of a WBEM-URI-UntypedInstancePath (as
 2397 defined in [DSP0207](#)); otherwise the value of the AlertingManagedElement property shall be Null.

2398 7.3.31.5.3 Property: AlertType

2399 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the AlertType property in
 2400 CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2401 content of the ALERT_TYPE element from the alert message definition in the message registry.

2402 7.3.31.5.4 Property: Message

2403 The requirement level of the Message property is optional.

2404 The Message property may contain the formatted alert message from the registry.

2405 **7.3.31.5.5 Property: MessageID**

2406 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the MessageID property in
 2407 CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2408 concatenation of the PREFIX and SEQUENCE_NUMBER attribute values from the alert message definition
 2409 in the message registry (that is, no further padding or adjustment of these values takes place).

2410 NOTE The SEQUENCE_NUMBER attribute value is not to be confused with the sequence number within a
 2411 sequence identifier that enables unique identification of the indications originating from a particular WBEM
 2412 server to a particular WBEM listener; see 7.4.2.

2413 **7.3.31.5.6 Property: OwningEntity**

2414 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the OwningEntity property
 2415 in CIM_AlertIndication instances conveying an alert message from a message registry to be set to the
 2416 content of the OWNING_ENTITY element from the alert message definition in the message registry.

2417 **7.3.31.5.7 Property: PerceivedSeverity**

2418 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the value of the PerceivedSeverity
 2419 property in CIM_AlertIndication instances conveying an alert message from a message registry to be set
 2420 to the content of the PERCEIVED_SEVERITY element from the alert message definition in the message
 2421 registry.

2422 **7.3.31.5.8 Property: SystemName**

2423 If the managed element for which the alert indication is reported is represented by a CIM instance within
 2424 the implementation, and the managed element is a component of a system that is represented by a
 2425 CIM_System instance, then the value of the SystemName property in the AlertIndication instance shall be
 2426 identical with the value of the Name property in the CIM_System instance; otherwise, the value of the
 2427 SystemName property shall be Null.

2428 **7.3.31.5.9 Property: MessageArguments[]**

2429 The requirements of [DSP0228](#) apply. Note that [DSP0228](#) requires the (string typed) MessageArguments
 2430 array property in CIM_AlertIndication instances conveying an alert message from a message registry to
 2431 contain one array entry for each dynamic element defined in the alert message, in the order specified by
 2432 the alert message definition in the message registry, where the value of the array element provides the
 2433 value of the dynamic element.

2434 If for a particular alert indication defined by a referencing profile the definition of a dynamic element
 2435 (including its description) within an alert message definition in a message registry is not sufficient to
 2436 identify a particular CIM instance and property as required by the referencing profile, then the referencing
 2437 profile shall specify augmenting provisions that explicitly identify an instance and a property that are
 2438 compatible with the definition of the dynamic element within the alert message.

2439 For example, assume that an alert message is defined in a message repository, as follows:

```
2440 <MESSAGE NAME="System state change">
2441   <MESSAGE_ID PREFIX="SVPC" SEQUENCE_NUMBER="0123" />
2442   <MESSAGE_DESCRIPTION>
2443     This message describes a system state change.
2444   </MESSAGE_DESCRIPTION>
2445   <MESSAGE_COMPONENTS>
2446     <STATIC_ELEMENT>The system </STATIC_ELEMENT>
2447     <DYNAMIC_ELEMENT NAME="SystemElementName"
2448       SOURCE_PROPERTY="CIM_System.ElementName" DATATYPE="string" />
2449     <STATIC_ELEMENT> changed its state to </STATIC_ELEMENT>
```

```

2450     <DYNAMIC_ELEMENT NAME="SystemState"
2451         SOURCE_PROPERTY="CIM_System.EnabledState" DATATYPE="string" />
2452     <STATIC_ELEMENT> . </STATIC_ELEMENT>
2453 </MESSAGE_COMPONENTS>
2454 <FIXED_MESSAGE_INSTANCE_VALUES TYPE="ALERT">
2455     <!-- . . . -->
2456 </FIXED_MESSAGE_INSTANCE_VALUES>
2457     <!-- . . . -->
2458 </MESSAGE>

```

2459 An Example System Virtualization profile might model an indication reporting state changes of both host
 2460 systems and virtual systems. In both cases the SVPC0123 alert message would be used, but the
 2461 identification of affected instances would need to be specialized separately for each case.

2462 Assuming that the profile defines a HostSystem adaptation of the CIM_System class for the
 2463 representation of host systems, and defines a HostStateChange indication adaptation in order to report
 2464 state changes of host systems, the requirements for the MessageArguments[] array property as part of
 2465 the HostStateChange indication adaptation would need to augment the alert message definition from the
 2466 message registry, as follows:

- 2467 • The value of MessageArguments[0] shall be the value of the ElementName property of the
 2468 HostSystem instance representing the host system that changed its state.
- 2469 • The value of MessageArguments[1] shall be the new value of the EnabledState property of the
 2470 HostSystem instance representing the host system that changed its state.

2471 7.3.31.6 Indication generation requirements

2472 The indication generation requirements of 7.3.29.5 apply respectively for the AlertIndication adaptation.

2473 7.3.32 LifecycleIndication: CIM_InstIndication

2474 7.3.32.1 General

2475 The LifecycleIndication adaptation models lifecycle indications of CIM instances; lifecycle indications are
 2476 described in 6.1.2.3.

2477 The LifecycleIndication adaptation adapts the CIM_InstIndication class and is based on the
 2478 BasicIndication adaptation (see 7.3.29); in addition, if the ReliableIndications feature (see 7.2.4) is
 2479 implemented, it is also based on the ReliableIndication adaptation (see 7.3.30).

2480 The implementation type of the LifecycleIndication indication adaptation is: "abstract".

2481 It is expected that the LifecycleIndication adaptation is used as a base adaptation for modeling lifecycle
 2482 indications in referencing profiles.

2483 7.3.32.2 Event definition requirements

2484 This subclause refines the event definition requirements established by the BasicIndication adaptation
 2485 (see 7.3.29.2) for the LifecycleIndication adaptation.

2486 Recall that lifecycle indication reports secondary events (see 6.1.1). The secondary event that is reported
 2487 by LifecycleIndication instances shall be described by an event definition query statement that conforms
 2488 to the following ABNF rule:

```

2489     "SELECT" WS PropertySet WS "FROM" WS LifecycleIndicationClass WS
2490     "WHERE" WS "ISA" WS ModelElement [ WS "WHERE" SelectionExpression ]

```

2491 PropertySet shall be "*", or a comma-separated list of property names.

2492 LifecycleIndicationClass shall be one of CIM_InstCreation, CIM_InstDeletion, or
 2493 CIM_InstModification, or a subclass of these indication classes.

2494 ModelElement shall identify a class for that the referencing profile defines a class adaptation, and for
 2495 which the modeled lifecycle indication reports secondary events. The class adaptation of that class shall
 2496 be stated as part of the description of the lifecycle indication adaptation in the referencing profile.

2497 NOTE For examples that comply with this requirement, see 7.3.33 and 7.3.34.

2498 SelectionExpression shall be a constant string that defines a selection expression conformant with
 2499 the rules for selection expressions defined by [DSP0202](#).

2500 NOTE These rules provide for referencing profiles being able to define one lifecycle indication for one target
 2501 adaptation per lifecycle indication adaptation. If for a particular target adaptation a referencing profile intends
 2502 to model lifecycle indications for different lifecycle events (such as the creation, destruction or modification
 2503 of instances of the target adaptation), for each of these lifecycle events separate lifecycle indication
 2504 adaptations are required. Furthermore, if lifecycle indications are to be modeled for different target
 2505 adaptations, for each target adaptation separate lifecycle indication adaptations are required. As usual, if
 2506 common requirements exist for such lifecycle indication adaptations, these can be defined in a common
 2507 abstract base adaptation that is used as a base for the specific lifecycle indication adaptations, thereby
 2508 avoiding the repetition of the commonalities.

2509 **7.3.32.3 Indication origin**

2510 The indication origin as required by 7.3.29.3 shall be the namespace of the CIM instance referenced by
 2511 the value of the SourceInstanceModelPath property (see 7.3.32.4.3).

2512 **7.3.32.4 Element requirements**

2513 **7.3.32.4.1 General**

2514 Table 47 lists the element requirements for the LifecycleIndication adaptation.

2515 **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.

2516 **7.3.32.4.2 Property: SourceInstance**

2517 The value of the SourceInstance property shall be an embedded instance of the class selected in the
 2518 query statement defining the event. The embedded instance shall be a copy of the instance for which the
 2519 lifecycle indication is reported. If the query statement specifies a specific selection of properties (other
 2520 than " * "), then the set of properties contained in the embedded instance shall be limited to those
 2521 selected; otherwise, the embedded instance shall at least contain values for each of the properties
 2522 required by the related adaptation of the selected class in the same referencing profile; see 7.3.29.2.

2523 **7.3.32.4.3 Property: SourceInstanceModelPath**

2524 The value of the SourceInstanceModelPath property shall refer to the same instance that is copied as an
 2525 embedded instance through the value of the SourceInstance property.

2526 **7.3.32.5 Indication generation requirements**

2527 The indication generation requirements of 7.3.29.5 apply respectively for the LifecycleIndication
 2528 adaptation.

2529 **7.3.33 ListenerDestinationRemovalIndication: CIM_InstDeletion**

2530 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2531 The ListenerDestinationRemovalIndication adaptation models a lifecycle indication that reports the
 2532 destruction of a CIM_ListenerDestination instance, as modeled in this profile by the ListenerDestination
 2533 adaptation (see 7.3.23). The destruction of a ListenerDestination instance is a secondary event caused
 2534 by the destruction of the represented listener destination; see 6.4.5.

2535 The requirement level of the ListenerDestinationRemovalIndication indication adaptation is optional.

2536 The implementation type of the ListenerDestinationRemovalIndication indication adaptation is:
 2537 "indication".

2538 Table 48 lists the element requirements for the ListenerDestinationRemovalIndication adaptation.

2539 **Table 48 – ListenerDestinationRemovalIndication: Element requirements**

Elements	Requirement	Description
Base adaptations		
LifecycleIndication	Mandatory	See 7.3.32.

2540 The requirement level of the ListenerDestinationRemovalIndication adaptation is optional.

2541 The event reported by the ListenerDestinationRemovalIndication adaptation is defined by the following
 2542 event definition query statement:

```
2543     SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA
2544     CIM_ListenerDestination
```

2545 **7.3.34 SubscriptionRemovalIndication: CIM_InstDeletion**

2546 The requirements in this subclause are WBEM server related implementation requirements; see 7.1.

2547 The SubscriptionRemovalIndication adaptation models a lifecycle indication that reports the destruction of
 2548 a CIM_AbstractIndicationSubscription instance, as modeled in this profile by the AbstractSubscription
 2549 adaptation (see 7.3.25). The destruction of a CIM_AbstractIndicationSubscription instance is a secondary
 2550 event caused by the destruction of the represented subscription; see 6.1.1.

2551 The requirement level of the SubscriptionRemovalIndication indication adaptation is optional.

2552 The implementation type of the SubscriptionRemovalIndication indication adaptation is: "indication".

2553 Table 49 lists the element requirements for the SubscriptionRemovalIndication adaptation.

2554

Table 49 – SubscriptionRemovalIndication: Element requirements

Elements	Requirement	Description
Base adaptations		
LifecycleIndication	Mandatory	See 7.3.32.

2555 The requirement level of the SubscriptionRemovalIndication adaptation is optional.

2556 The event reported by the SubscriptionRemovalIndication adaptation is defined by the following query
2557 statement:

```
2558     SELECT * FROM CIM_InstDeletion WHERE SourceInstance ISA
2559     CIM_AbstractIndicationSubscription
```

2560 **7.4 Reliable indication delivery**

2561 **7.4.1 General**

2562 This subclause defines mechanisms for the reliable delivery of indications from an implementation to a
2563 listener as described in 6.1.5.

2564 Implementations implementing the ReliableIndications feature (see 7.2.4) shall comply with the
2565 requirements specified in 7.4.3; note that in addition the requirements of the ReliableIndications
2566 adaptation (see 7.3.30) apply.

2567 Implementations not implementing the ReliableIndications feature are not required to comply with the
2568 provisions in this subclause or those in 7.3.30.

2569 Listeners implementing the ReliableIndications feature (see 7.2.4) shall comply with the provisions stated
2570 in 7.4.4. Listeners not implementing the ReliableIndications feature are not required to comply with these
2571 provisions and may ignore the sequence identifiers in received indications, as exposed by the values of
2572 the SequenceContext and SequenceNumber properties in any received CIM_Indication instances.

2573 **7.4.2 Sequence identifier and sequence identifier lifetime**

2574 This subclause defines the concepts of *sequence identifier* and *sequence identifier lifetime*.

2575 The *sequence identifier* within an indication enables unique identification of the indications originating
2576 from a particular WBEM server to a particular WBEM listener.

2577 A sequence identifier is composed of a sequence context and a sequence number.

2578 NOTE The sequence number within a sequence identifier is not to be confused with the SEQUENCE_NUMBER
2579 attribute value that is part of the identification of the alert message that defines an alert indication; see
2580 7.3.31.5.5.

2581 The sequence context is required to be unique for each listener destination maintained by the indication
2582 service within a WBEM server; within that context the sequence number is required to be unique for each
2583 indication delivered from the WBEM server to the listener referenced by the listener destination. The
2584 requirements for the CIM representation of the sequence identifier in reliable indications are defined in
2585 7.3.30.

2586 The *sequence identifier lifetime* maintained by an implementation is a duration defined as follows:

2587
$$\text{sequence-identifier-lifetime} = \text{number-of-retry-attempts} * \text{delivery-retry-interval} * 10$$

2588 In this formula the number-of-retry-attempts is the number of retry attempts as indicated by the value of
2589 the DeliveryRetryAttempts property (see 7.3.2.3.3) in the IndicationService instance representing the

2590 indication service within the implementation, and the delivery-retry-interval is the duration of the delivery
2591 retry interval as indicated by the value of the DeliveryRetryInterval property (see 7.3.2.3.4) in the same
2592 instance.

2593 Within the sequence identifier lifetime an implementation that is implementing reliable indications may
2594 attempt to retry failed indication delivery attempts, as detailed in 7.4.3, and a listener implementing
2595 reliable indications may expect the delivery of anticipated indications, as detailed in 7.4.4.

2596 **7.4.3 WBEM server requirements**

2597 **7.4.3.1 General**

2598 Indication delivery is based on a publish/subscribe event paradigm, where an implementation delivers
2599 indications to subscribed listeners. The indication delivery may fail for various reasons, including
2600 unavailability of the listener or network issues. This subclause describes the requirements for the
2601 implementation that are related to reliable indication delivery. The mechanisms to deliver indications and
2602 to determine success or failure of indication delivery are protocol dependent; see the specifications of
2603 applicable protocols that specify mechanisms for indication delivery.

2604 **7.4.3.2 Prohibition of indication delivery for disabled or removed subscriptions**

2605 If a subscription is disabled or has been removed, the implementation should discard any undelivered
2606 indications for that subscription. For example, this applies if the implementation has queued indications
2607 for delivery retry, and the subscription is removed by a client before the delivery retry is executed.

2608 **7.4.3.3 Prohibition of repeated indication delivery**

2609 After an implementation has successfully delivered an indication to a listener, it shall not deliver that
2610 indication again to that same listener.

2611 **7.4.3.4 Requirements for the retry of failed indication deliveries**

2612 If the attempt to deliver an indication to a particular listener fails, the implementation shall retry the
2613 indication delivery as detailed in this subclause.

- 2614 1) The implementation shall wait for the duration of the delivery retry interval, as exposed by the
2615 value of the DeliveryRetryInterval property in the IndicationService instance (see 7.3.2)
2616 representing the indication service within the implementation.
- 2617 2) If the actual number of retry attempts is less than the maximum number of retry attempts as
2618 exposed by the value of the DeliveryRetryAttempts property in the IndicationService instance
2619 representing the indication service within the implementation, and the elapsed time after the first
2620 delivery is less than the sequence identifier lifetime as defined in 7.4.2, the implementation shall
2621 retry the failed indication delivery.
- 2622 • If the retry is successful, delivery of that indication to the particular listener is complete.
 - 2623 • If the retry is not successful, and preconditions of step 2) still apply, then the
2624 implementation shall re-iterate starting with step 1).
 - 2625 • Otherwise, the indication shall be considered as not deliverable to the particular listener,
2626 and the requirements defined in 7.4.3.5 apply.

2627 **7.4.3.5 Requirements for undeliverable indications**

2628 This subclause defines the implementation behavior if an indication has been considered unable to be
2629 delivered to a listener, as described in 7.4.3.4.

2630 If the listener destination referencing that listener is permanent (see 7.3.23.3.3), the implementation shall
2631 record an error and shall no longer attempt to deliver that indication to that listener (that is, the
2632 implementation shall discard it). This action does not modify the listener destination and any of its
2633 subscriptions.

2634 If the listener destination referencing that listener is transient (see 7.3.23.3.3), the implementation shall
2635 record an error and shall no longer attempt to deliver that indication to that listener (that is, the
2636 implementation shall discard it). In addition, the listener destination and its subscriptions may be removed
2637 from the implementation as described in 7.4.3.6.

2638 **7.4.3.6 Requirements for the implicit removal of subscriptions and listener destinations**

2639 An implementation may remove a subscription and the referenced listener destination if the delivery of
2640 one or more indications to the represented listener failed as described in 7.4.3.4 and 7.4.3.5.

2641 The implementation behavior with respect to the implicit removal of subscriptions and listener destinations
2642 shall be exposed by the value of the SubscriptionRemovalAction property in the IndicationService
2643 instance representing the responsible indication service; see 7.3.2.3.5.

2644 **7.4.3.7 Behavior related to WBEM server restarts**

2645 Indications that have been generated but not yet delivered may get lost during a WBEM server crash or
2646 restart because there is not requirement to persist such indications.

2647 If the implementation chooses an algorithm for the construction of the sequence context part of the
2648 sequence identifier (see 7.4.2) that includes the WBEM server startup time, the potential re-use of the
2649 same sequence identifier is implicitly avoided. That way listeners can deal with indication delivery failures
2650 caused by WBEM server restarts in the same way they deal with other kinds of indication delivery failures.

2651 **7.4.4 WBEM listener requirements**

2652 **7.4.4.1 General**

2653 A listener shall keep track of each distinct sequence identifier of any indications received from a particular
2654 indication service for the duration of the sequence identifier lifetime maintained by that indication service,
2655 counting from the last time that sequence identifier was detected in a received indication from that
2656 indication service. If the same sequence identifier is used by two different indication services (for
2657 example, in two different implementations), the listener shall keep track of them independently.

2658 After the lifetime of a sequence identifier expires, the listener should discard the knowledge about that
2659 sequence identifier from that indication service. After the knowledge about a sequence identifier for an
2660 indication service has been discarded by the listener, a new usage of that sequence identifier in an
2661 indication from that indication service shall be treated by the listener like a new, unknown sequence
2662 identifier from that indication service.

2663 Keeping track of sequence identifiers in listeners enables the detection of lost and duplicate deliveries,
2664 and the detection and re-ordering of indications arriving out of order, as described in 7.4.4.5. Discarding
2665 the knowledge about sequence identifiers minimizes the resource requirements of the listener.

2666 **7.4.4.2 Determination of the expected sequence identifier of the next indication**

2667 From the sequence identifier of the last indication received from a particular implementation, a listener
2668 shall infer the expected sequence identifier of the next indication by incrementing the sequence number
2669 by 1, wrapping to an initial value of 0 if the maximum limit has been reached, and maintaining the
2670 sequence context.

2671 7.4.4.3 Lost indications

2672 If the sequence identifier of the next received indication sent from the same implementation does not
2673 match the expected value as described in 7.4.4.2, the listener shall consider the expected indication as a
2674 candidate for a lost indication. After waiting for the sequence identifier lifetime period as maintained by
2675 the implementation sending that indication, the listener shall conclude that the expected indication is lost.

2676 7.4.4.4 Duplicate indications

2677 Any additional indications received from the same implementation with the same sequence identifier shall
2678 be considered duplicates. In this case, the lifetime for the sequence identifier shall be adjusted starting
2679 with the delivery time of the most recently received duplicate indication, and adding the sequence
2680 identifier lifetime period as maintained by the implementation sending that indication.

2681 7.4.4.5 Out-of-order indications

2682 A listener that intends to re-establish the original order of indications before processing them needs to
2683 defer the processing of any prematurely arriving indication that does not have the expected sequence
2684 number, until the decision can be made as to whether the expected indications are lost.

2685 If the sequence identifier of the next received indication does not match the expected sequence identifier
2686 as described in 7.4.4.2, the listener shall cache such prematurely arriving indications and wait for delivery
2687 of the indication with the expected sequence identifier for a period of time defined by the sequence
2688 identifier lifetime (as defined in 7.4.4.1) of the last received indication from the same implementation.

2689 If the indication with the expected sequence identifier is not received during that period, the expected
2690 indication should be considered lost (see 7.4.4.3).

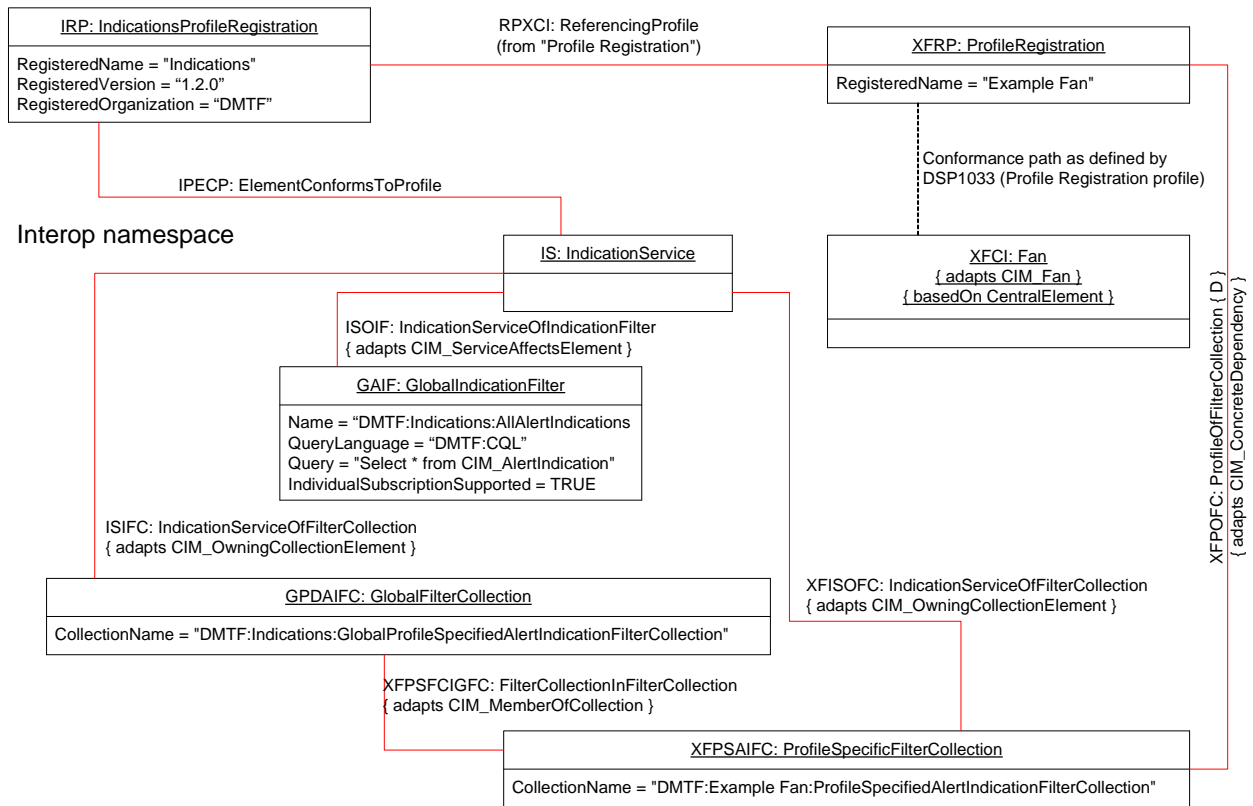
2691 If the indication with the expected sequence identifier is received during that period, the indication order
2692 shall be re-ordered using their sequence numbers, such that the indications are processed in the order
2693 they were sent by the implementation.

2694 **8 Use Cases**

2695 **8.1 Object Diagrams**

2696 Figure 4 depicts a DMTF object diagram. It shows CIM instances exposed by the implementation of an
 2697 Example Fan profile that defines some indications (not shown in the diagram), and thus is required by
 2698 [DSP1001](#) to reference this profile, implying the implementation of respective elements defined in this
 2699 profile.

2700



2701

2702 **Figure 4 – DMTF object diagram: Global and profile-specific filter collections**

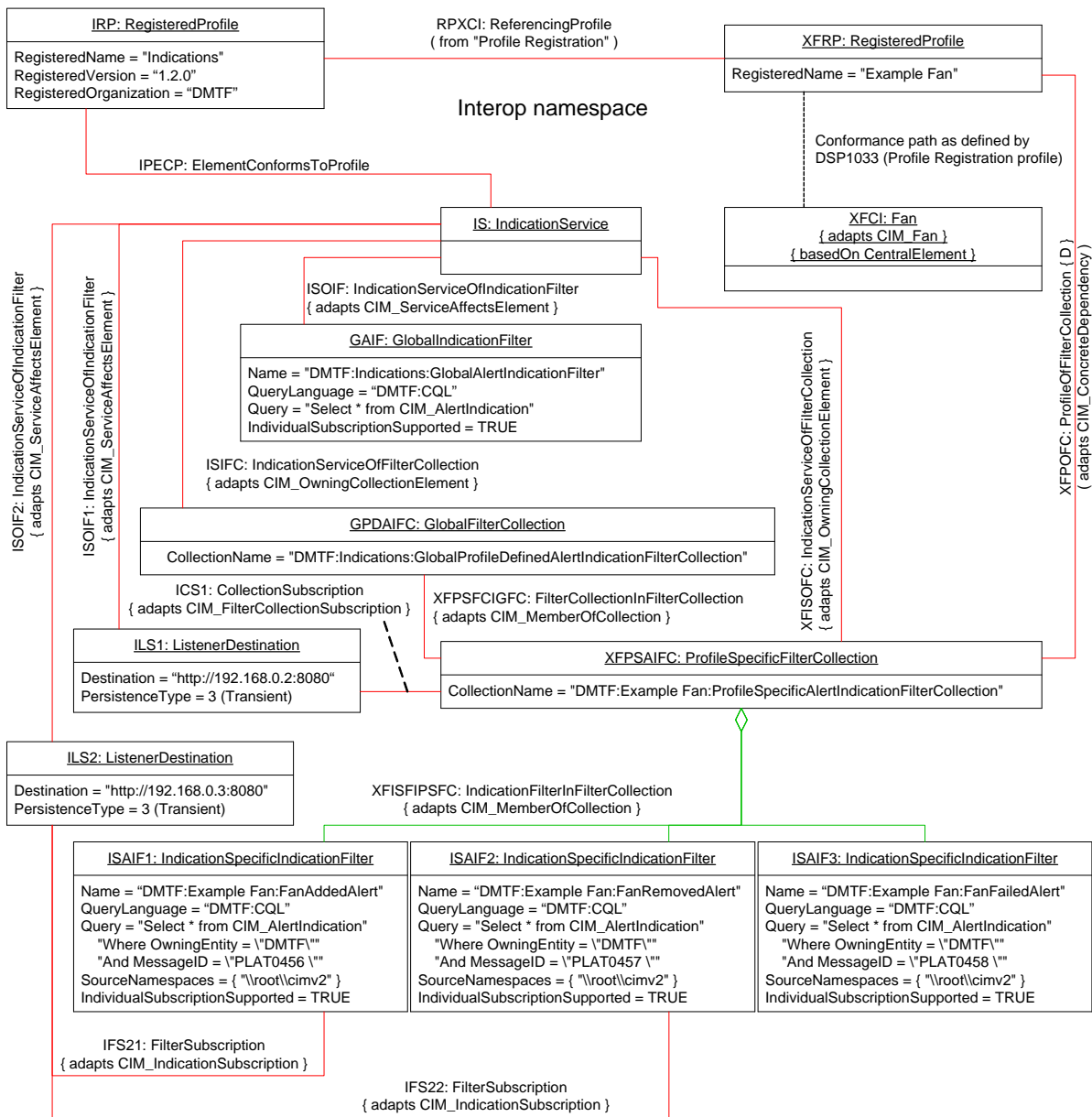
2703 The implemented version of this profile is represented by the RegisteredProfile instance IRP, the
 2704 implemented version of the Example Fan profile is represented by RegisteredProfile instance XFRP, and
 2705 the reference relationship is shown by the ReferencingProfile association instance RPXCI.

2706 The implementation of this profile exposes the IndicationService (see 7.3.2) instance IS representing the
 2707 implemented indication service. It also exposes the GlobalIndicationFilter (see 7.3.16) instance GAIF
 2708 representing the global indication filter covering all alert indications.

2709 Furthermore, the implementation of this profile exposes the GlobalFilterCollection (see 7.3.22) instance
 2710 GPDAIFC representing the global filter collection for alert indications with a defined coverage covering all
 2711 profile defined alert indications. The implementation of the Example Fan profile exposes the
 2712 ProfileSpecificFilterCollection (see 7.3.21) instance XFPSAIFC representing the related profile-specific
 2713 filter collection for alert indications with a defined coverage covering all alert indications defined in the
 2714 Example Fan profile.

2715 The global filter collection for alert indications represented by GPDAIFC contains the profile-specific filter
2716 collection for alert indications represented by XFPSAIFC; this containment relationship is represented by
2717 the FilterCollectionInFilterCollection (see 7.3.20) instance XFPSFCIGFC. Because the coverage of the
2718 global filter collection is explicitly represented by containment, in this case its coverage is inspectable by
2719 clients. However, the CIM representation of the contained profile-specific filter collection for alert
2720 indications represented by XFPSAIFC does not expose any contained elements. In that case clients
2721 would require prior knowledge of the defined coverage, that is, all alert indications defined in the Example
2722 Fan profile, which (because of the explicitly represented containment relationship) is in this example also
2723 the coverage of the global filter collection for alert indications represented by GPDAIFC.

2724 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)

2725

2726

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

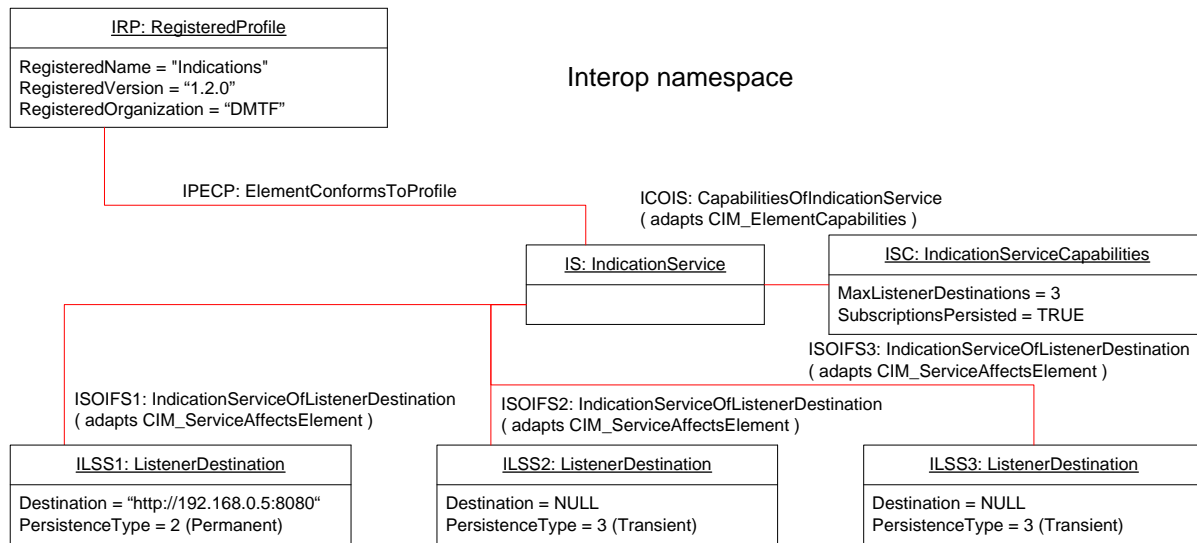
2727 The first difference from the situation shown in Figure 4 is that in Figure 5 the profile-specific filter
 2728 collection for alert indications represented by XFISAIFC contains three indication filters, represented by
 2729 the IndicationSpecificIndicationFilter instances ISAIF1, ISAIF2 and ISAIF3. Hence the coverage of the
 2730 profile-specific filter collection for alert indications represented by XFPSAIFC is now defined by the
 2731 contained indication filters, that is, it covers the three alert indications described by the alert messages
 2732 with the IDs PLAT0456, PLAT0457, and PLAT0458.

2733 It is important to recapture that — as with any indication gate — the presence of the CIM representation
 2734 of specific indication filters does not indicate that the covered indications are actually implemented. The
 2735 semantics of indication gates are defined with respect to *filtering*, but *not* with respect to *generating*,
 2736 indications (see 7.3.11.2 and 7.3.17.2). Thus, a subscribed listener is guaranteed only to be delivered any
 2737 *generated* indication that is within the coverage of the indication gate, but the *generation* of the indication
 2738 is not guaranteed. For that reason referencing profiles need to model other elements — such as
 2739 capabilities — for the purpose of conveying the information about which indications defined in the
 2740 referencing profile are actually implemented and thus generated when the respective event occurs; the
 2741 definition of such mechanisms is outside the scope of this profile.

2742 The second difference between Figure 4 and Figure 5 is that in Figure 5 listener destinations are
 2743 represented by the ListenerDestination instances ILS1 and ILS2. The listener referenced by ILS1 is
 2744 subscribed to the profile-specific filter collection represented by XFPSAIFC, and the listener referenced
 2745 by ILS1 is subscribed to the indication-specific indication filters represented by ISAIF1 and ISAIF2.

2746 Lastly, the representations of three indications are shown at the bottom of Figure 5, along with their origin
 2747 namespace. Each of these indications is within the coverage of the indication filter represented directly
 2748 above it. Thus, the alert indications represented by XFALERT1 and XFALERT2 are delivered to both the
 2749 listeners represented by ILS1 and ILS2, whereas XFALERT3 is only delivered to ILS1.

2750 Figure 6 depicts the DMTF object diagram for an implementation that supports a fixed number of listener
 2751 destinations.



2752

2753

Figure 6 – DMTF object diagram: Static listener destinations

2754 In the example shown in Figure 6, an implementation supports a maximum of three listener destinations,
 2755 indicated by the value of the MaxListenerDestinations property in the IndicationServiceCapabilities
 2756 instance ISC that describes the capabilities of the indication service within the implementation. The three
 2757 listener destinations are represented by the three respective ListenerDestination instances ILSS1, ILSS2,
 2758 and ILSS3. The listener destination represented by ILSS1 is currently configured as a permanent listener
 2759 destination, referencing the listener reachable under URI "http://192.168.0.5:8080". The listener

2760 destinations represented by ILSS2 and ILSS3 currently are free listener destinations as indicated by the
 2761 value Null for the Destination property, that is, they are not currently configured for a specific listener. A
 2762 client can request modifications of any of the listener destinations in order to reference a desired listener
 2763 for indication delivery by modifying the representing ListenerDestination instances.

2764 **8.2 LocateIndicationService: Locate the indication service provided by an** 2765 **implementation of this profile**

2766 **8.2.1 Preconditions**

2767 The client knows the following:

- 2768 • The identifying information of a WBEM server (for example, its IP address and the port number
 2769 if the WBEM server implements CIM operations over http as described in [DSP0223](#))
- 2770 • Name, required version, and registered organization of this profile as stated in 7.3.5

2771 **8.2.2 Flow of activities**

- 2772 1) The client obtains all IndicationsProfileRegistration instances (see 7.3.5), applying respective
 2773 use cases described in [DSP1033](#) to locate CIM_RegisteredProfile instances representing profile
 2774 registrations of particular profiles and selecting those instances where the values of the
 2775 RegisteredName, RegisteredVersion, and RegisteredOrganization properties match the
 2776 required input values.

2777 The result is zero or more IndicationsProfileRegistration instances (see 7.3.23).

2778 NOTE 1 Typically only one instance is returned, but if this profile is implemented more than once within the
 2779 identified WBEM server, more than one instance may be returned.

2780 If no instance was detected, this use case is complete and the client knows that the required
 2781 version of this profile is not implemented within the WBEM server. If one or more instances
 2782 were detected, any of them represents the required version of this profile, and the client can
 2783 select any of these for further processing.

- 2784 2) The client applies use cases described in [DSP1033](#) in order to locate instances of the
 2785 IndicationService adaptation that is the central class adaptation defined in this profile.

2786 The result is zero or one IndicationService instances (see 7.3.2).

2787 NOTE 2 Technically, more than one instance could be returned, but that would indicate a non-compliant
 2788 implementation of this profile.

2789 If no instance was detected, this use case is complete and the client knows that an indication
 2790 service is not presently active within the identified WBEM server. If one or more instances were
 2791 detected, any of them represents an indication service compliant to the requirements specified
 2792 in this profile, and the client can select any of these for further processing.

2793 **8.2.3 Postconditions**

2794 Unless errors occurred, the client either knows an IndicationService instance (including its object path)
 2795 representing an indication service within the identified WBEM server with a behavior compliant to the
 2796 requirements specified in this profile or knows that either this profile is not implemented within the
 2797 identified WBEM server or that no indication service is presently active within the identified WBEM server.

2798 **8.3 LocateProfileIndicationService: Locate the indication service responsible for**
 2799 **delivering indications defined by a referencing profile**

2800 **8.3.1 Preconditions**

2801 The client knows the following:

- 2802 • The ProfileRegistration instance (including its object path) representing the profile registration of
 2803 the referencing profile

2804 **8.3.2 Flow of activities**

- 2805 1) For the input ProfileRegistration instance, find the IndicationsProfileRegistration instances (see
 2806 7.3.5) associated through ReferencedProfile instances (see [DSP1033](#)) (for example, using the
 2807 GetAssociatedInstancesWithPath() operation).

2808 The result is zero or one IndicationsProfileRegistration instances (see 7.3.5).

2809 NOTE 1 Technically, more than one instance could be returned, but that would indicate a non-compliant
 2810 implementation of the referencing profile.

2811 If no instance was detected, this use case is complete and the client knows that the
 2812 implementation of the referencing profile did not implement indications.

- 2813 2) For the IndicationsProfileRegistration instance obtained in step 1), find the IndicationService
 2814 instances (see 7.3.2) associated through ElementConformsToProfile instances (see 7.3.6) (for
 2815 example, using the GetAssociatedInstancesWithPath() operation).

2816 The result is zero or one IndicationService instances (see 7.3.2).

2817 NOTE 2 Technically, more than one instance could be returned, but that would indicate a non-compliant
 2818 implementation of this profile.

2819 **8.3.3 Postconditions**

2820 Unless errors occurred, the client knows an IndicationService instance (including its object path)
 2821 representing an indication service that is responsible for delivering indications defined by the referencing
 2822 profile.

2823 **8.4 DetermineIndicationServiceCapabilities: Determine the capabilities of an**
 2824 **indication service**

2825 **8.4.1 Preconditions**

2826 The client knows all of the following:

- 2827 • a copy of the IndicationService instance (including its object path) representing the indication
 2828 service within the implementation

2829 NOTE For example, that IndicationService instance could be obtained by applying the LocateIndicationService
 2830 use case (see 8.2) or the LocateProfileIndicationService use case (see 8.3).

2831 **8.4.2 Flow of activities**

- 2832 1) Inspecting property values of the IndicationService instance (see 7.3.2.3), the client can already
 2833 determine some aspects of the behavior of the represented indication service.

2834 For example, the value of the FilterCreationEnabled property indicates whether the support for
 2835 dynamic indication filters as modeled by the DynamicIndicationFilters feature (see 7.2.1) is
 2836 available.

2837 The values of the DeliveryRetryAttempts, the DeliveryRetryInterval, the
 2838 SubscriptionRemovalAction, and the SubscriptionRemovalTimeInterval indicate if and to what
 2839 extent the support for reliable indications as modeled by the ReliableIndications feature (see
 2840 7.2.4) is available.

2841 2) Find the IndicationsServiceCapabilities instance (see 7.3.7) representing the capabilities of the
 2842 input indication service, by traversing the CIM_ServiceAffectsElement association modeled by
 2843 the CapabilitiesOfIndicationService association adaptation (see 7.3.8) by invoking the
 2844 GetAssociatedInstancesWithPath() operation with the following actual values for the input
 2845 parameters:

- 2846 – InstanceName: the object path to the input IndicationService instance
- 2847 – AssocClass: "CIM_ElementCapabilities", the adapted class of the
 2848 CapabilitiesOfIndicationService association adaptation
- 2849 – ResultClass: "CIM_IndicationServiceCapabilities", the adapted class of the
 2850 IndicationServiceCapabilities adaptation

2851 The result is zero or one IndicationServiceCapabilities instance.

2852 NOTE Technically, more than one instance could be returned, but that would indicate a non-compliant
 2853 implementation of this profile.

2854 If an IndicationServiceCapabilities instance was returned, the use case continues with step 3);
 2855 otherwise, it continues with step 4).

2856 3) Inspect the property values of the returned IndicationServiceCapabilities instance (see 7.3.7).
 2857 The values of those properties with names ending with "IsSettable" enable the client to
 2858 determine whether client modification of respective aspects of the behavior of the input
 2859 indication service is possible. The values of the MaxListenerDestinations and the
 2860 MaxActiveSubscriptions properties expose the upper limits for the number of listener
 2861 destinations and for the number of subscriptions supported by the indication service, and the
 2862 value of the SubscriptionsPersisted property exposes whether subscriptions are persisted over
 2863 restarts of the input indication service. This step completes this use case.

2864 4) Continue here after step 2) if no IndicationServiceCapabilities instance was returned. In this
 2865 case, client modification of the indication service is not supported, and the upper limits for the
 2866 number of supported listener destinations and subscriptions is not exposed by the
 2867 implementation; in addition, whether subscriptions are persisted over indication service restarts
 2868 is not exposed.

2869 8.4.3 Postconditions

2870 Unless errors occurred, the client knows the capabilities of the input indication service as far as it is
 2871 exposed by the representing IndicationService instance, by the related IndicationServiceCapabilities
 2872 instance, and by initial behavior specified in this profile.

2873 8.5 ModifyIndicationService: Modify functional aspects of an indication service

2874 The client knows all of the following:

- 2875 • a copy of the IndicationService instance (including its object path) (see 7.3.2) representing the
 2876 indication service within the implementation (see the LocateIndicationService use case in 8.2)

- 2877 • a copy of the IndicationServiceCapabilities instance (including its object path) (see 7.3.7)
 2878 representing the capabilities of the indication service within the implementation (See the
 2879 DetermineIndicationServiceCapabilities use case in 8.4.)

2880 **8.5.1 Flow of activities**

- 2881 1) Inspect the property values in the input IndicationsServiceCapabilities instance (see 7.3.7)
 2882 representing the capabilities of the input indication service to determine which properties in the
 2883 IndicationService instance are modifiable. (See step 3) in the
 2884 DetermineIndicationServiceCapabilities use case in 8.4.)
- 2885 2) If admissible by the determination of step 1), in the input local copy of the input
 2886 IndicationService instance, modify property values as desired. For example, if the value of the
 2887 DeliveryRetryAttemptsIsSettable property in the IndicationServiceCapabilities instance is True,
 2888 a modification of the corresponding DeliveryRetryAttempts property in the IndicationService
 2889 instance is admissible.
- 2890 3) Use the ModifyInstance() operation to request the desired change in the behavior of the
 2891 indication service, providing the modified copy of the IndicationService instance as the actual
 2892 value of the ModifiedInstance parameter.

2893 **8.5.2 Postconditions**

2894 Unless errors occurred, the desired change of functional aspects of the input indication service is
 2895 effective.

2896 **8.6 ListListenerDestinations: List all listener destinations exposed by an** 2897 **implementation**

2898 **8.6.1 Preconditions**

2899 The client knows all of the following:

- 2900 • the object path to the IndicationService instance representing the indication service within the
 2901 implementation (see 8.2)

2902 **8.6.2 Flow of activities**

- 2903 1) Find all listener destinations within the responsibility of the indication service by traversing the
 2904 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfListenerDestination
 2905 adaptation (see 7.3.24) by invoking the GetAssociatedInstancesWithPath() operation with the
 2906 following actual values for the input parameters:
- 2907 – InstanceName: the object path to the input IndicationService instance
 - 2908 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
 2909 IndicationServiceOfListenerDestination adaptation
 - 2910 – ResultClass: "CIM_ListenerDestination", the adapted class of the ListenerDestination
 2911 adaptation
- 2912 The result is a set of ListenerDestination instances (see 7.3.23).

2913 **8.6.3 Postconditions**

2914 Unless errors occurred, the client knows all ListenerDestination instances (including their object paths)
 2915 representing all the listener destinations maintained by the implementation.

2916 **8.7 SelectListenerDestination: Select an existing listener destination referencing**
 2917 **a desired listener**

2918 **8.7.1 Preconditions**

2919 The client knows all of the following:

- 2920 • the object path to the IndicationService instance representing the indication service within the
2921 implementation (see 8.2)
- 2922 • the URI exposed by the desired listener
- 2923 • the particular protocol to be applied when delivering these indications

2924 **8.7.2 Flow of activities**

2925 1) Execute the ListListenerDestinations use case (see 8.6).

2926 The result is a set of ListenerDestination instances (see 7.3.23).

2927 2) Inspect each ListenerDestination instance resulting from step 1) by checking the value of the
2928 Destination property against the input URI, and by checking whether the value of the Protocol
2929 property matches the particular protocol for this use case.

2930 If both conditions are met, the located ListenerDestination represents a listener destination that
2931 within the implementation represents the particular listener, and this use case is complete;
2932 otherwise, the client needs to repeat step 2), inspecting further ListenerDestination instances
2933 from the result of step 1).

2934 3) If all result elements from step 1) checked in step 2) did not yield a ListenerDestination instance
2935 referencing the listener, then this use case is complete and the client knows that the listener is
2936 not presently represented by a listener destination within the implementation.

2937 **8.7.3 Postconditions**

2938 Unless errors occurred, the client either knows a ListenerDestination instance (including its object path)
2939 representing a listener destination within the implementation that references the particular listener, or
2940 knows that the listener is not referenced by any listener destination within the implementation.

2941 In the latter case, and if the implementation has also implemented the dynamic creation of listener
2942 destinations, the client could apply the CreateListenerDestination use case (see 8.8) to dynamically
2943 create a respective listener destination within the implementation that represents the desired listener.

2944 **8.8 CreateListenerDestination: Create a new listener destination**

2945 **8.8.1 Preconditions**

2946 The client knows all of the following:

- 2947 • The same as for the SelectListenerDestination use case; see 8.7.1.

2948 **8.8.2 Flow of activities**

2949 1) Execute the SelectIndicationFilter use case (see 8.7).

2950 If a listener destination referencing the desired listener is found, use that; in this case, this use
2951 case is complete.

- 2952 2) Prepare a local instance of the CIM_ListenerDestination class that complies with the
 2953 requirements of the ListenerDestination adaptation (see 7.3.23), inserting property values as
 2954 follows:
- 2955 – Destination: the identification of the listener that the new listener destination is to
 2956 reference, using the format required in 7.3.23.3.2. The format needs to be compatible
 2957 with the requested protocol.
 - 2958 – PersistenceType: the durability requested for the new listener destination, using the
 2959 format required in 7.3.23.3.3.
 - 2960 – Protocol: the protocol to used for the communication with the listener, using the format
 2961 required by the CIM schema definition of the CIM_ListenerDestination class.
- 2962 3) Request the creation of the new listener destination in the implementation by invoking the
 2963 CreateInstance() operation, providing the CIM_ListenerDestination instance prepared in step 2)
 2964 as the actual value of the NewInstance parameter.
- 2965 If successful, the operation returns the object path of the ListenerDestination instance
 2966 representing the newly created listener destination.
- 2967 If not successful, the operation returns a CIM status code providing details about the failure
 2968 (see 7.3.23.3.4).

2969 8.8.3 Postconditions

2970 Unless errors occurred, the client knows the object path of a ListenerDestination instance representing a
 2971 listener destination referencing the desired listener that either preexisted or was created; otherwise, the
 2972 client knows details about why it was not possible to find or dynamically create the respective listener
 2973 destination.

2974 8.9 FindFreeListenerDestination: Find a free listener destination

2975 8.9.1 Preconditions

2976 The client knows all of the following:

- 2977 • the object path to the IndicationService instance representing the indication service within the
 2978 implementation (see 8.2)

2979 8.9.2 Flow of activities

- 2980 1) Execute the ListListenerDestinations use case (see 8.6).
- 2981 The result of this step is the set of ListenerDestination instances (including their object paths)
 2982 representing all the listener destinations within the implementation.
- 2983 2) From the result of step 1), select a free listener destination; free listener destinations are
 2984 represented by those ListenerDestination instances where the value of the Destination property
 2985 is Null.

2986 8.9.3 Postconditions

2987 Unless errors occurred, the client knows a free listener destination, or knows that presently no free
 2988 listener destinations exist within the implementation.

2989 **8.10 ModifyListenerDestination: Modify an existing listener destination**

2990 **8.10.1 Preconditions**

2991 The client knows all of the following:

- 2992 • a local copy of a ListenerDestination instance (see 7.3.23)

2993 NOTE For example, the listener destination and its representing ListenerDestination instance might have been
2994 obtained by executing the FindFreeListenerDestination use case described in 8.9.

2995 **8.10.2 Flow of activities**

- 2996 1) Modify the local copy of the ListenerDestination instance, maintaining compliance with the
2997 requirements of the ListenerDestination adaptation (see 7.3.23).
- 2998 2) Modify the listener destination maintained by the implementation by invoking the
2999 ModifyInstance() operation, providing the CIM_ListenerDestination instance prepared in step 1)
3000 as the actual value of the ModifiedInstance parameter.

3001 If successful, the operation returns without error; otherwise, the operation returns a CIM status
3002 code providing details about the failure (see 7.3.23.3.6).

3003 **8.10.3 Postconditions**

3004 Unless errors occurred, the listener destination represented by the input ListenerDestination instance was
3005 modified; otherwise, the client knows details about why it was not possible to modify the represented
3006 listener destination.

3007 **8.11 DeleteListenerDestination: Delete an existing listener destination**

3008 **8.11.1 Preconditions**

3009 The client knows all of the following:

- 3010 • the object path to a ListenerDestination instance (see 7.3.23)

3011 **8.11.2 Flow of activities**

- 3012 1) For the input ListenerDestination instance, find all AbstractSubscription instances (see 7.3.25)
3013 referencing the ListenerDestination instance (for example, using the
3014 GetReferencingInstancePaths() operation).
- 3015 2) Delete all subscriptions referencing the input listener destination by executing the
3016 DeleteSubscription use case (see 8.21) for each AbstractSubscription instance returned by step
3017 1).
- 3018 3) Invoke the DeleteInstance() operation on the input ListenerDestination instance, effecting the
3019 deletion of the referenced listener destination.

3020 **8.11.3 Postconditions**

3021 Unless errors occurred, the input listener destination is deleted and no longer represented by any
3022 ListenerDestination instances.

3023 8.12 FindIndicationFilter: Find an indication filter covering a particular indication

3024 8.12.1 Preconditions

3025 The client knows all of the following:

- 3026 • the object path to the IndicationService instance representing the indication service within the
3027 implementation (see 7.3.2)
- 3028 • an implemented indication. Knowledge about whether or not a particular indication is actually
3029 implemented could for example be obtained by inspecting respective capabilities exposed by an
3030 implementation of a referencing profile that defines an adaptation of the particular indication.

3031 8.12.2 Flow of activities

3032 1) Find all indication filters within the responsibility of the indication service by traversing the
3033 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfIndicationFilter
3034 association adaptation (see 7.3.14) by invoking the GetAssociatedInstancesWithPath()
3035 operation with the following actual values for the input parameters:

- 3036 – InstanceName: the object path to the input IndicationService instance
- 3037 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
3038 IndicationServiceOfIndicationFilter association adaptation
- 3039 – ResultClass: "CIM_IndicationFilter", the adapted class of the IndicationFilter adaptation

3040 The result of this step is a set of IndicationFilter instances (see 7.3.11).

3041 2) Inspect each IndicationFilter instance resulting from step 1) by first checking the value of the
3042 QueryLanguage property. If the query language indicated by that value is interpretable by the
3043 client, interpret the query statement presented by the value of the Query property; otherwise,
3044 continue inspecting the next IndicationFilter instance returned by step 1).

3045 If the desired indication is not within the coverage as expressed by the query statement, then
3046 continue inspecting the next IndicationFilter instance returned by step 1).

3047 3) If the client desires to subscribe to the indication filter, continue by inspecting the IndicationFilter
3048 instance resulting from step 1) by checking whether the value of the
3049 IndividualSubscriptionSupported property is True. If so, this use case is complete; otherwise,
3050 continue with step 2) inspecting the next IndicationFilter instance returned by step 1); otherwise,
3051 this use case is complete.

3052 8.12.3 Postconditions

3053 Unless errors occurred, and if step 3) produced a suitable IndicationFilter instance, the client by that
3054 instance (including its object path) knows an indication filter that covers the desired indication and that
3055 supports individual subscriptions; otherwise, the client knows that within the responsibility of the indication
3056 service no such indication filter exists.

3057 8.13 DetermineQueryLanguages: Determine the set of query languages 3058 supported for query statements

3059 8.13.1 Preconditions

3060 The client knows all of the following:

- 3061 • The same as for the FindIndicationFilter use case described in 8.12.1.

3062 NOTE The procedure outlined in this use case is only an auxiliary approach to be pursued if preliminary
3063 knowledge about the query languages supported by an implementation is not available to the client.

3064 **8.13.2 Flow of activities**

- 3065 1) Execute steps 1) and 2) of the FindIndicationFilter use case (see 8.9), but vary step 2) to collect
 3066 the query languages applied by all the inspected indication filters.

3067 **8.13.3 Postconditions**

3068 Unless errors occurred, the client knows all the query languages in use by existing indication filters.

3069 NOTE Because not all query languages supported by an implementation might be in use by indication filters, the
 3070 set of query languages obtained by executing this use case is actually an open subset of the set of
 3071 supported query languages.

3072 **8.14 CreateIndicationFilter: Create a dynamic indication filter covering a**
 3073 **particular indication**

3074 **8.14.1 Preconditions**

3075 The client knows all of the following:

- 3076 • The same as for the FindIndicationFilter use case described in 8.12.1.

3077 **8.14.2 Flow of activities**

- 3078 1) Execute the FindIndicationFilter use case (see 8.9).

3079 If a suitable indication filter covering the desired indication is found, use that; in this case, this
 3080 use case is complete.

- 3081 2) If not already done previously, execute step 1) of the DetermineIndicationServiceCapabilities
 3082 use case (see 8.4) and determine by the value of the FilterCreationEnabled property whether
 3083 the support for dynamic indication filters as modeled by the DynamicIndicationFilters feature
 3084 (see 7.2.1) is available.

- 3085 3) If the set of query languages supported by the implementation is not known a priori, execute the
 3086 DetermineQueryLanguages use case (see 8.13).

- 3087 4) Prepare a local instance of the CIM_IndicationFilter class that complies with the requirements of
 3088 the DynamicIndicationFilter adaptation (see 7.3.13), inserting property values as follows:

- 3089 – QueryLanguage: a query language supported by the implementation; see 7.3.11.3.6.
- 3090 – Query: the query statement covering the desired set of indications; see 7.3.11.3.5.

3091 NOTE Additional constraints on properties of the CIM_Indication class selected by the
 3092 query statement may be specified through the WHERE clause; however, if the
 3093 implementation is unable to comply with these constraints, the operation will fail.

- 3094 – SourceNamespaces[]: a list of local namespace names identifying the namespaces
 3095 considered as ; see 7.3.11.3.3.

- 3096 5) Request the creation of the new dynamic indication filter in the implementation by invoking the
 3097 CreateInstance() operation, providing the CIM_IndicationFilter instance prepared in step 4) as
 3098 the actual value of the NewInstance parameter.

3099 If successful, the operation returns the object path of the DynamicIndicationFilter instance
 3100 representing the newly created dynamic indication filter.

3101 If not successful, the operation returns a CIM status code providing details about the failure
 3102 (see 7.3.13.2.2).

3103 8.14.3 Postconditions

3104 Unless errors occurred, the client knows the object path of an IndicationFilter instance representing an
3105 indication filter covering the desired indication that either preexisted or was dynamically created;
3106 otherwise, the client knows details about why it was not possible to find or dynamically create the
3107 respective indication filter.

3108 8.15 ModifyIndicationFilter: Modify a dynamic indication filter**3109 8.15.1 Preconditions**

3110 The client knows all of the following:

- 3111 • a local copy of an DynamicIndicationFilter instance (see 7.3.13)

3112 NOTE For example, that dynamic indication filter and its representing DynamicIndicationFilter instance might
3113 have been created by executing the CreateIndicationFilter use case; see 8.14.

3114 8.15.2 Flow of activities

- 3115 1) Modify the local copy of the DynamicIndicationFilter instance, maintaining compliance with the
3116 requirements of the DynamicIndicationFilter adaptation (see 7.3.13).
- 3117 2) Modify the dynamic indication filter maintained by the implementation by invoking the
3118 ModifyInstance() operation, providing the DynamicIndicationFilter instance prepared in step 1)
3119 as the actual value of the ModifiedInstance parameter.
- 3120 3) If successful, the operation returns without error; otherwise, the operation returns a CIM status
3121 code providing details about the failure (see 7.3.13.2.4).

3122 8.15.3 Postconditions

3123 Unless errors occurred, the dynamic indication filter represented by the input DynamicIndicationFilter
3124 instance was modified; otherwise, the client knows details about why it was not possible to modify the
3125 represented dynamic indication filter.

3126 8.16 DeleteIndicationFilter: Delete a dynamic indication filter**3127 8.16.1 Preconditions**

3128 The client knows all of the following:

- 3129 • the object path to a DynamicIndicationFilter instance (see 7.3.13)

3130 8.16.2 Flow of activities

- 3131 1) For the input DynamicIndicationFilter instance, find all AbstractSubscription instances (see
3132 7.3.25) referencing the DynamicIndicationFilter instance (for example, using the
3133 GetReferencingInstancePaths() operation).
- 3134 2) Delete all subscriptions referencing the input listener destination, by executing the
3135 DeleteSubscription use case (see 8.21) for each AbstractSubscription instance returned by step
3136 1).
- 3137 3) Invoke the DeleteInstance() operation on the input DynamicIndicationFilter instance, effecting
3138 the deletion of the referenced dynamic indication filter.

3139 **8.16.3 Postconditions**

3140 Unless errors occurred, the input dynamic indication filter is deleted and no longer represented by any
3141 DynamicIndicationFilter instances.

3142 **8.17 CheckCollectionCoverage: Check the coverage of a filter collection**3143 **8.17.1 Preconditions**

3144 The client knows all of the following:

- 3145 • a local copy of a StaticFilterCollection instance (see 7.3.17), and the object path referencing the
3146 original StaticFilterCollection instance within the implementation

3147 **8.17.2 Flow of activities**

3148 1) Check whether the input filter collection contains any elements by resolving — from the
3149 StaticFilterCollection instance — the CIM_ConcreteComponent association as modeled by the
3150 IndicationFilterInFilterCollection association adaptation (see 7.3.19) and the
3151 FilterCollectionInFilterCollection association adaptation (see 7.3.20).

3152 If no contained elements are discovered, a defined coverage may apply as the coverage; in this
3153 case, skip to step 4).

3154 2) For each of the contained elements found in step 1), determine the contributed coverage and
3155 add that to the resulting aggregated coverage of the input filter collection.

3156 In the case of a contained indication filter, the contributed coverage is determined by inspecting
3157 the values of the QueryLanguage property and that of the Query property containing the query
3158 statement.

3159 In the case of a contained filter collection, the contributed coverage is determined by recursively
3160 applying this use case (8.17).

3161 3) Aggregate the contributed coverage of each contained element as determined in step 2) into the
3162 resulting aggregated coverage of the input filter collection. After completing this step the client
3163 knows the aggregated coverage of the input filter collection, and this use case is complete.

3164 4) This step applies if no contained elements were discovered in steps 2) and 3).

3165 Check the value of the CollectionName property in the StaticFilterCollection instance for the
3166 pattern required for the name the global filter collection covering all instance lifecycle
3167 indications, as detailed in 7.3.22.4.4.

3168 If the pattern matches, the client knows that the represented filter collection is the global filter
3169 collection covering all instance lifecycle indications; in this case, the client knows that the
3170 coverage of the input filter collection is all instance lifecycle indications and this use case is
3171 complete.

3172 5) Check the value of the CollectionName property in the StaticFilterCollection instance for the
3173 pattern required for the name of global filter collections for profile defined indications, as defined
3174 in 7.3.22.

3175 If the pattern matches, the client knows that the represented filter collection is a global filter
3176 collection for profile defined indications with a defined coverage as detailed in 7.3.22. The client
3177 needs to have a priori knowledge about the defined coverage of each referencing profile, and
3178 this use case is complete.

3179 6) Check the value of the CollectionName property in the StaticFilterCollection instance for the
3180 pattern required for the name of profile-specific filter collections as defined in 7.3.21.2.2.

3181 If the pattern matches, the client knows that the input filter collection is a profile-specific filter
3182 collection with a defined coverage as detailed in 7.3.21.3. The client needs to have a priori
3183 knowledge about the defined coverage of the identified referencing profile, and this use case is
3184 complete.

3185 7) If the input filter collection does not match any of the types determined in steps 4), 5), and 6),
3186 then no defined coverage applies. Furthermore, because no contained elements were
3187 discovered in step 2), the coverage of the input filter collection is empty (that is, it does not
3188 cover any indications).

3189 **8.17.3 Postconditions**

3190 Unless errors occurred, or in the cases determined in steps 5) and 6) above the client does not have a
3191 priori knowledge about the defined coverage(s), the client knows the coverage of the input filter collection.

3192 **8.18 ObtainNamedCollection: Obtain a named filter collection**

3193 **8.18.1 Preconditions**

3194 The client knows all of the following:

- 3195 • the object path to the IndicationService instance representing the indication service within the
3196 implementation (see 7.3.2)
- 3197 • the name of the named filter collection, for example, the name of a global filter collection or of a
3198 profile-specific filter collection

3199 **8.18.2 Flow of activities**

3200 1) Find all filter collections within the responsibility of the indication service by traversing the
3201 CIM_ServiceAffectsElement association modeled by the IndicationServiceOfFilterCollection
3202 association adaptation (see 7.3.18) by invoking the GetAssociatedInstancesWithPath()
3203 operation with the following actual values for the input parameters:

- 3204 – InstanceName: the object path to the input IndicationService instance
- 3205 – AssocClass: "CIM_ServiceAffectsElement", the adapted class of the
3206 IndicationServiceOfFilterCollection association adaptation
- 3207 – ResultClass: "CIM_FilterCollection", the adapted class of the StaticFilterCollection
3208 adaptation

3209 The result of this step is a set of StaticFilterCollection instances (see 7.3.17).

3210 2) Inspect each StaticFilterCollection instance resulting from step 1) by checking the value of the
3211 CollectionName property. If the name of the static filter collection as indicated by that value
3212 matches the desired name, this use case is complete; otherwise, continue inspecting the next
3213 IndicationFilter instance returned by step 1).

3214 **8.18.3 Postconditions**

3215 Unless errors occurred, the client knows the named filter collection by means of the representing
3216 StaticFilterCollection instance (including its object path).

3217 **8.19 CreateSubscription: Create a subscription**

3218 **8.19.1 Preconditions**

3219 The client knows all of the following:

- 3220 • the object path to the IndicationService instance representing the indication service within the
3221 implementation (see 7.3.2)
- 3222 • an object path to an IndicationFilter instance representing an indication filter covering the
3223 desired indication or set of indications
- 3224 For example, see the FindIndicationFilter (8.12) or CreateIndicationFilter (8.14) use cases about
3225 how to obtain that object path.
- 3226 • Alternatively, an object path to a StaticFilterCollection instance representing a filter collection
3227 covering the desired indication or set of indications. For example, see the
3228 ObtainNamedCollection use case (8.18) about how to obtain the object path to a
3229 StaticFilterCollection instance representing a global filter collection or a profile-specific filter
3230 collection.
- 3231 • an object path to a ListenerDestination instance representing a listener destination that
3232 represents the desired listener within the implementation. For example, see the
3233 SelectListenerDestination use case (8.7) about how to obtain that object path.

3234 8.19.2 Flow of activities

- 3235 1) Prepare a local instance of the CIM_IndicationSubscription class (or the
3236 CIM_FilterCollectionSubscription for a subscription to a filter collection) that complies with the
3237 requirements of the FilterSubscription adaptation (see 7.3.26) or the CollectionSubscription
3238 adaptation (see 7.3.27), inserting property values as follows:
 - 3239 – Filter: input object path to the indication filter (or to the filter collection)
 - 3240 – Handler: input object path to the listener destination
- 3241 The values of other properties should be specified in conformance with the capabilities of the
3242 implementation as exposed by instances of the IndicationService adaptation and the
3243 IndicationServiceCapabilities adaptation; see the DetermineIndicationServiceCapabilities use
3244 case (8.4) to obtain knowledge about these capabilities.
- 3245 Values not described through these adaptations may or may not be respected by the
3246 implementation; in this case it is implementation dependent whether in step 2) the
3247 implementation imposes a respective default behavior, or whether it fails in creating the new
3248 subscription.
- 3249 2) Define the new subscription to the implementation by invoking the CreateInstance() operation,
3250 providing the CIM_IndicationSubscription (or CIM_FilterCollectionSubscription) instance
3251 prepared in step 1) as the actual value of the NewInstance parameter.
- 3252 If successful, the operation returns the object path of the DynamicIndicationFilter instance
3253 representing the newly created subscription.
- 3254 If not successful, the operation returns a CIM status code providing details about the failure
3255 (see 7.3.26.3.2 or 7.3.27.3.2).

3256 8.19.3 Postconditions

- 3257 Unless errors occurred, the client knows the object path of an AbstractSubscription instance representing
3258 the newly created subscription; otherwise, the client knows details about why it was not possible to create
3259 the subscription.

3260 **8.20 CheckSubscriptions: Determine whether subscriptions exist for a given**
3261 **indication and listener**

3262 **8.20.1 Preconditions**

3263 The client knows all of the following:

- 3264 • the object path to the IndicationService instance representing the indication service within the
3265 implementation (see 8.2)
- 3266 • the URI exposed by the desired listener

3267 **8.20.2 Flow of activities**

- 3268 1) Execute the ListListenerDestinations use case (see 8.6). The result is a set of
3269 ListenerDestination instances (including their object paths) representing all the listener
3270 destinations within the implementation.
 - 3271 2) From the result of step 1), drop all ListenerDestination instances not referencing the desired
3272 listener. The result is a set of ListenerDestination instances (including their object paths)
3273 representing all the listener destinations referencing the desired listener.
 - 3274 3) For each ListenerDestination instance resulting from step 2), find all IndicationFilter instances
3275 (see 7.3.11) associated with the ListenerDestination instance (see 7.3.23) through a
3276 FilterSubscription instance (see 7.3.26). The result of this step is a set of IndicationFilter
3277 instances representing indication filters to which the desired listener is subscribed.
 - 3278 4) Inspect each IndicationFilter instance resulting from step 3) by checking the values of the
3279 QueryLanguage and the Query properties. Interpret the query statement expressed by the value
3280 of the Query property and check whether the input indication is covered. If the input indication is
3281 covered, add the identification of the represented listener destination to a filter result list, and
3282 continue inspecting the next IndicationFilter instance returned by step 3).
 - 3283 5) For each ListenerDestination instance resulting from step 2), find all StaticFilterCollection
3284 instances (see 7.3.17) associated through a CollectionSubscription instance (see 7.3.27). The
3285 result of this step is a set of StaticFilterCollection instances representing static filter collections
3286 to which the desired listener is subscribed.
 - 3287 6) For each StaticFilterCollection instance resulting from step 5), apply the
3288 CheckCollectionCoverage use case (see 8.17).
- 3289 If the input indication is covered, add the identification of the represented static filter collection to
3290 a collection result list, and continue inspecting the next StaticFilterCollection instance returned
3291 by step 5).

3292 **8.20.3 Postconditions**

3293 Unless errors occurred, the client knows (the identifications of) all listener destinations and filter
3294 collections to which the desired listener is subscribed.

3295 **8.21 DeleteSubscription: Delete a subscription**

3296 **8.21.1 Preconditions**

3297 The client knows all of the following:

- 3298 • the object path to the AbstractSubscription instance (see 7.3.25) representing a subscription
3299 within the implementation

3300 **8.21.2 Flow of activities**

3301 1) Invoke the DeleteInstance() operation on the AbstractSubscription instance, effecting the
3302 deletion of the represented subscription.

3303 NOTE If the subscription referenced a dynamic indication filter, and no other subscriptions reference it, and the
3304 client does not plan to create a new subscription for this filter, the client can delete the dynamic indication
3305 filter using the DeleteFilter use case (see 8.16); likewise, unless referenced by other subscriptions, the
3306 client can delete the listener destination that was referenced by the deleted subscription, using the
3307 DeleteListenerDestination use case (see 8.11).

3308 **8.21.3 Postconditions**

3309 Unless errors occurred, the subscription is deleted and no longer represented by any
3310 AbstractSubscription instance.

3311 **8.22 FindAlertingSystem: Find the system containing a component causing an
3312 alert indication**3313 **8.22.1 Preconditions**

3314 The client knows all of the following:

- 3315 • an AlertIndication instance representing an alert indication that references the alerting managed
3316 element

3317 **8.22.2 Flow of activities**

3318 1) Obtain the CIM element referenced by the value of the AlertingManagedElement in the input
3319 AlertIndication instance.

3320 2) Determine the profile with which the CIM element is conformant and where the central class
3321 adaption adapts the CIM_System class.

3322 NOTE This step implies client knowledge about profiles defining adaptations of the class of the CIM
3323 element obtained in step 1). More than one profile could impact the CIM element, but the
3324 scoping CIM_System instance should be the same in all cases.

3325 3) Use the scoping algorithm defined by the profile determined in step 2) to find the related
3326 instance of the scoping class adaptation of that profile.

3327 **8.22.3 Postconditions**

3328 Unless errors occurred, the client knows the CIM_System instance representing the system containing a
3329 component causing the generation of the input alert indication.

3330 **8.23 DetermineIndicationGate: Determine the indication gate of an indication**3331 **8.23.1 Preconditions**

3332 The client knows all of the following:

- 3333 • an AlertIndication instance representing an alert indication that references the alerting managed
3334 element

3335 In addition, subscriptions for the listener that received the input alert indication should have been
3336 established such that within the set of subscribed to indication gates within a particular implementation
3337 each is uniquely identified with a name as exposed by the value of the Name property in representing

3338 IndicationFilter instances (see 7.3.11), or as exposed by the value of the CollectionName property in
 3339 representing StaticFilterCollection instances (see 7.3.17).

3340 NOTE This policy ensures that indication gate names are unique with respect to one implementation;
 3341 implementations are unable to (and not required to) maintain that uniqueness, but clients can ensure it
 3342 through carefully applying the subscription policy stated above for each listener that a client controls.

3343 8.23.2 Flow of activities

3344 1) Extract the value of the IndicationFilterName from the input AlertIndication instance as the name
 3345 of the sought-after indication gate.

3346 If the input alert indication originates from an implementation that is known to the client by
 3347 reference to its representing IndicationFilter instance, skip to step 8); otherwise, continue with
 3348 step 2).

3349 2) Inspect the value of the AlertingManagedElement property of the input AlertIndication instance.

3350 If that value is Null, then the indication gate cannot be determined, and this use case is
 3351 complete without success; this is also the case of the value is a URI that does not reference a
 3352 CIM instance that represents the alerting managed element. In subsequent steps it is assumed
 3353 that the value is a URI that references a CIM instance that represents the alerting managed
 3354 element.

3355 3) Determine the ProfileRegistration instance that is providing the CIM instance referenced by the
 3356 URI found in step 2), using one of the algorithms described in [DSP1033](#) for that purpose.

3357 4) Apply the LocateProfileIndicationService use case (see 8.3) in order to determine the
 3358 IndicationService instance (see 7.3.2) that represents the indication service from which the input
 3359 alert indication originated.

3360 5) Find all IndicationFilter instances (see 7.3.11) associated with the IndicationFilter instance (see
 3361 7.3.23) found in step 4) through an IndicationServiceOfIndicationFilter instance (see 7.3.14), for
 3362 example by executing the GetAssociatedInstancesWithPath() operation.

3363 6) For each IndicationFilter instance obtained in step 5), determine if the value of the Name
 3364 property matches the name of the sought-after indication gate determined in step 1).

3365 If it matches, and the subscription policy mentioned in the preconditions was maintained, then
 3366 the indication filter represented by the IndicationFilter instance is the sought-after indication
 3367 gate.

3368 If the name matches, and the subscription policy was not maintained, then all IndicationFilter
 3369 instances determined in step 5) need to be checked with step 6) in order to ensure that the
 3370 name as exposed by the value of the Name property is not used more than once. If this is the
 3371 case, the sought-after indication gate cannot be exactly determined; however, at least it can be
 3372 limited to the set of indication filters using the name as determined in step 1).

3373 If a name does match, continue with step 8).

3374 If the name does not match, the next instance from the set determined in step 5) needs to be
 3375 checked with step 6); if no additional instances remain, continue with step 7).

3376 7) Repeat steps 5) and 6) for filter collections, searching for StaticFilterCollection instances (see
 3377 7.3.17) associated through an IndicationServiceOfFilterCollection instance (see 7.3.18) in step
 3378 5), and checking the value of the CollectionName property in step 6).

3379 8) If an indication filter was determined as the sought-after indication gate in steps 1), 6), or 7), the
 3380 client can check the query statement exposed by the value of the Query property in the
 3381 representing IndicationFilter instance (or — in case the alert indication was received through a
 3382 filter collection — in at least one of the contained IndicationFilter instances), and verify that the

3383 input alert indication is indeed within the coverage of the identified indication filter or filter
3384 collection.

3385 8.23.3 Postconditions

3386 Unless errors occurred, the client knows the indication gate emitting the input alert indication by means of
3387 its representing IndicationFilter or StaticFilterCollection instance.

3388 8.24 SubscribeForProfileIndications: Subscribe for all of the indications defined 3389 in a referencing profile

3390 8.24.1 Preconditions

3391 The client knows the following:

- 3392 • the registered name of the referencing profile
- 3393 • the object path to the IndicationService instance representing the indication service within the
3394 implementation (see 7.3.2)
- 3395 • the object path to the ListenerDestination instance (see 7.3.23) representing the desired listener
3396 destination

3397 8.24.2 Flow of activities

- 3398 1) Construct the name for the profile-specific filter collection for alert indications, applying the
3399 pattern defined in 7.3.21.2.2.
- 3400 2) Execute the ObtainNamedCollection use case (see 8.18), providing the name constructed in
3401 step 1) as input; the result is either Null or the object path referencing the
3402 ProfileSpecificAlertIndicationFilterCollection instance (see 7.3.21) representing the profile-
3403 specific filter collection for alert indications of the referencing profile.
- 3404 3) If an object path was returned on step 2), execute the CreateSubscription use case (see 8.19),
3405 providing that object path and the input object path to the ListenerDestination instance as input.
- 3406 4) Perform steps 1), 2) and 3) analogously for lifecycle indications.

3407 8.24.3 Postconditions

3408 Unless errors occurred, the desired listener destination is subscribed for all alert indications and all
3409 lifecycle indications defined by the referencing profile.

3410

ANNEX A (informative)

3411
3412
3413
3414

Profiles defining indications

3415 Referencing profiles define indications and related requirements in the following ways:

- 3416 • Reference this profile as a mandatory or conditional profile
- 3417 • Define lifecycle indications and/or alert indications by defining adaptations based on the
3418 LifecycleIndication adaptation (see 7.3.32) and/or the AlertIndication adaptation (see 7.3.31).
3419 This requires but is not limited to defining the requirement level, the reported event, and the
3420 query statement; however, the latter two may be implied by the respective base adaptation.
- 3421 • Optionally, define indication filters by defining adaptations based on the StaticIndicationFilter
3422 adaptation (see 7.3.11). The definition of indication-specific indication filters covering each
3423 lifecycle indication and each alert indication defined in a referencing profile is implied by this
3424 profile through the IndicationSpecificIndicationFilter adaptation (see 7.3.15), but may be refined
3425 by referencing profiles.
- 3426 • Optionally, define filter collections by defining adaptations based on the StaticFilterCollection
3427 adaptation (see 7.3.17). The definition of profile-specific filter collections covering all lifecycle
3428 indications and/or alert indications defined in a referencing profile is implied by this profile
3429 through the ProfileSpecificFilterCollection adaptation (see 7.3.21), but may be refined by
3430 referencing profiles.

**ANNEX B
(informative)**

Change Log

3431
3432
3433
3434

Version	Date	Description
1.0.0a	2007-06-04	Preliminary Standard
1.0.0	2008-12-05	Final Standard
1.0.1	2009-09-07	Released as DMTF Standard, with the following changes: <ul style="list-style-type: none"> • Updated profile conventions for operations and their usage • Fixed incorrect CIM Schema version (from 2.16 to 2.22)
1.1.0a	2009-12-02	Released as Work in Progress, with the following changes: <ul style="list-style-type: none"> • Increased CIM Schema version to 2.23(exp). • Added support for reliable indications (delivery retry, detection of lost indications, reconstruction of original order): <ul style="list-style-type: none"> – Description of reliable indications concept in 7.10 (Indication Delivery). – Clarifications in description of CIM_ListenerDestination.PersistenceType. • Refined the format for CIM_FilterCollection.CollectionName in 7.6. • Refined the format for CIM_IndicationFilter.Name in 7.4. • Cleaned up terminology clause by removing most terms that are defined in DSP0004, DSP0200 or DSP1001. • Added "Document conventions" clause and consolidated existing text into that. • Updated profile conventions for operations to match DSP1001 1.0.1. • Fixed incorrect pattern value "WBEMURI" for CIM_AlertIndication.AlertingElementFormat.
1.1.0	2010-05-20	Released as DMTF Standard, with the following changes: <ul style="list-style-type: none"> • Clarified and added some terms in clause 3. • Clarified that there is only one indication service in a WBEM server, but added a recommendation for clients to expect more than one in the future. • Fixed incorrect verbiage of sending indications to clients, to sending indications to listeners. • Changed ambiguous "conditional/optional" requirement to "conditional or optional" in all cases but one. • Clarified that listeners that intend to re-establish the original order of indications need to buffer indications that do not have the predicted sequence number until decision about loss can be made. • Lowered the requirement not to interpret sequence numbers in case of not implementing them, to a permission to ignore them. • Fixed inconsistencies in several diagrams.

Version	Date	Description
1.2.0a	2010-06-16	<p>Released as Work in Progress, with the following changes:</p> <ul style="list-style-type: none"> • Increased CIM Schema version to 2.25 • Converted to PUG 1.1 "Condensed Format": <ul style="list-style-type: none"> – The semantics of the definitions from the previous version was maintained, except the semantical changes detailed – Introduced separation between managed environment and CIM model – Defined many new terms precisely capturing concepts only vaguely defined in the previous version – Introduced features – Introduced adaptations, integrating the content of the Methods and the "CIM elements" clauses defined in the previous version into the "Implementation" clause of this version – Modified existing use cases using adaptations, and introduced new use cases • Introduced the following new concepts: <ul style="list-style-type: none"> – Global filters – Global filter collections – Profile-specific filters – Profile-specific filter collections • Deprecated the use of the CIM_ConcreteDependency association for modeling the relationship between filter collections and profile representations of referencing profiles (CIM_RegisteredProfile) • Changed the requirement level for the ElementConformsToProfile association adaptation to mandatory • Fixed incorrect property name: CIM_ListenerDestination.Protocol was incorrectly named ProtocolType. • Many clarifications of existing concepts, such as the following: <ul style="list-style-type: none"> – Established indication emitters as the super type of indication filters and filter collections – Clarified that the purpose of indication emitters is filtering indications, and is not the representation of indication implementations – Restructured the specification of reliable indications using a feature and adaptations – Consistently use the terms "sequence identifier" and "sequence identifier lifetime", as established by the CIM schema (quit using the term "sequence identifier value") – Suppression of repeated indication delivery

Version	Date	Description
1.2.0b	2010-09-15	<p>Released as Work in Progress, with the following changes:</p> <ul style="list-style-type: none"> • Included cPubs major scrub • Renamed indication emitter -> indication gate • Renamed profile-specific filter -> indication-specific filter • Removed specializations of these (introduced in the 1.2.0a version) • Deprecate requiring a CIM_Error instance in case of IndicationFilter.CreateInstance() error • Recommending the Interop namespace as the only namespace for IndicationFilter instances, StaticFilterCollection instances, ListenerDestination instances and AbstractSubscription instances • Many clarifications of existing concepts and addition of new concepts, such as the following: <ul style="list-style-type: none"> – Require that all kinds of filters have one or more related namespaces, either those identified by the value of the SourceNamespaces[] property, or – if the value is Null - the namespace where the filter representation resides; version 1.1 left that open for dynamic filters. – Prohibit empty array as possible SourceNamespaces[] value, as that would be semantically useless because no indication would be allowed to pass in this case. – Requiring that indications have an origin namespace – Requiring that the origin namespace is taken into consideration during indication filtering, i.e., is subject to filtering. This is done by extending the concept of the filter coverage such that both query statement and the namespace list span the filter coverage – Correcting the prohibition of providing any key properties when creating dynamic indication filters by exempting the Name property, along with a recommended naming convention

Version	Date	Description
1.2.0c	2011-04-05	<p>Released as a DMTF Draft Standard, with the following changes:</p> <ul style="list-style-type: none"> • Adjusted to the DMTF Draft Standard version of DSP1001 1.1 <ul style="list-style-type: none"> – Moved base elements from the table of class adaptations to the individual element requirements tables of the adaptations – Adopted the format for error reporting requirements – Require Key properties to be listed when used the first time in a chain of adaptations – Introduction of the implementation type – Restructured the error reporting requirement tables • Minor corrections resulting from reviews of version 1.2.0b • Adjust the use of the Profile Registration profile to DSP1033 1.0 • Specify operation requirements in terms of DSP0223 (as required by DSP1001, after Architecture workgroup decision) • Rephrased the policies for the avoidance of repeated indication delivery, synchronizing it with the phraseology used in the schema description of the CIM_AbstractIndicationSubscription class • Resolved various comments from 2nd SNIA review • Changed the requirement level of IndicationServiceOfIndicationFilter, IndicationServiceOfFilterCollection, CollectionSubscription and ProfileOfFilterCollection from conditional to mandatory because the condition was always true (the GlobalFilter and GlobalFilterCollection adaptations are mandatory derived adaptation of the IndicationFilter and StaticFilterCollection adaptations) • Reinforced the version 1.1 requirement that key properties on the creation of DynamicIndicationFilter instances are to be ignored, and should not be provided by clients • Extended the AlertIndication adaptation to allow for referencing more than one alert message • Extended the IndicationSpecificIndicationFilter adaptation to provide for multiple instances for the coverage of multi-message AlertIndication adaptations
1.2.0	2011-06-30	<p>Released as a DMTF Standard, with the following changes:</p> <ul style="list-style-type: none"> • Confirmed the CIM schema definition of CIM_Indication wrt. that a sequence identifier needs to be maintained on a per listener destination basis (and not on a per listener basis)

3435