



1

2

3

4

Document Number: DSP1018

Date: 2011-06-30

Version: 1.1.0

5 **Service Processor Profile**

6 **Document Type: Specification**

7 **Document Status: DMTF Standard**

8 **Document Language: en-US**

9

10 Copyright Notice

11 Copyright © 2006–2007, 2011 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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26 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

CONTENTS

33	Foreword	7
34	Introduction	8
35	1 Scope	9
36	2 Normative References.....	9
37	2.1 Approved References	9
38	2.2 Other References.....	10
39	3 Terms and Definitions	10
40	4 Symbols and Abbreviated Terms.....	12
41	5 Synopsis.....	12
42	6 Description	13
43	7 Implementation.....	14
44	7.1 Representing a Service Processor	14
45	7.2 Modeling Service Processor Redundancy (Optional).....	17
46	7.3 Managing Service Processor Time (Optional)	18
47	7.4 User Account Management (Optional)	18
48	7.5 Boot Control Profile (Optional).....	18
49	7.6 CLP Service Profile (Optional).....	18
50	7.7 DHCP Client Profile (Optional)	18
51	7.8 DNS Client Profile (Optional)	18
52	7.9 Ethernet Port Profile (Optional).....	18
53	7.10 Software Inventory Profile (Optional).....	18
54	7.11 Software Update Profile (Optional).....	19
55	7.12 IP Interface Profile (Optional)	19
56	7.13 Physical Asset Profile (Optional)	19
57	7.14 Record Log Profile (Optional)	19
58	7.15 Sensors Profile (Optional).....	19
59	7.16 Power State Management Profile (Optional)	19
60	7.17 Shared Device Management Profile (Optional)	19
61	7.18 SMASH Collections Profile (Optional)	19
62	7.19 SSH Service Profile (Optional)	19
63	7.20 Telnet Service Profile (Optional).....	20
64	7.21 Text Console Redirection Profile (Optional)	20
65	7.22 PCI Device Profile (Optional).....	20
66	8 Methods.....	20
67	8.1 Method: CIM_ComputerSystem.RequestStateChange()	20
68	8.2 Method: CIM_RedundancySet.Failover()	21
69	8.3 Method: CIM_TimeService.ManageTime()	22
70	8.4 Profile Conventions for Operations.....	23
71	8.5 CIM_ComputerSystem.....	23
72	8.6 CIM_HostedService	24
73	8.7 CIM_IsSpare	24
74	8.8 CIM_ElementCapabilities	25
75	8.9 CIM_EnabledLogicalElementCapabilities.....	25
76	8.10 CIM_MemberOfCollection	25
77	8.11 CIM_RedundancySet.....	25
78	8.12 CIM_TimeService	25
79	8.13 CIM_ServiceAffectsElement	26
80	9 Use Cases.....	26
81	9.1 Object Diagrams	26
82	9.2 Reset a Service Processor	29
83	9.3 Retrieve the Service Processor Redundancy Status.....	30
84	9.4 Determine Whether Manual Failover Is Supported	30

85	9.5	Force a Service Processor Failover.....	30
86	9.6	Determine Whether the ElementName Is Modifiable	30
87	9.7	Determining If State Management Is Supported	30
88	10	CIM Elements	31
89	10.1	CIM_ComputerSystem.....	31
90	10.2	CIM_ElementCapabilities	32
91	10.3	CIM_EnabledLogicalElementCapabilities.....	32
92	10.4	CIM_HostedService	32
93	10.5	CIM_IsSpare	33
94	10.6	CIM_MemberOfCollection	33
95	10.7	CIM_OwningCollectionElement.....	33
96	10.8	CIM_RedundancySet.....	34
97	10.9	CIM_RegisteredProfile.....	34
98	10.10	CIM_ServiceAffectsElement	34
99	10.11	CIM_TimeService	35
100	10.12	CIM_ManagementController.....	35
101		ANNEX A (informative) Change Log.....	36
102			

103 Figures

104	Figure 1 – Service Processor Profile: Class Diagram.....	14
105	Figure 2 – Base Server	26
106	Figure 3 – Modular System.....	27
107	Figure 4 – Service Processors before Failover.....	28
108	Figure 5 – Service Processors after Failover.....	29
109		

110 Tables

111	Table 1 – Referenced Profiles	12
112	Table 2 – CIM_ComputerSystem.EnabledState Value Description.....	15
113	Table 3 – CIM_ComputerSystem.RequestStateChange() Method: Return Code Values.....	21
114	Table 4 – CIM_ComputerSystem.RequestStateChange() Method: Parameters.....	21
115	Table 5 – CIM_RedundancySet.Failover() Method: Return Code Values.....	22
116	Table 6 – CIM_RedundancySet.Failover() Method: Parameters.....	22
117	Table 7 – CIM_TimeService.ManageTime() Method: Return Code Values	23
118	Table 8 – CIM_TimeService.ManageTime() Method: Parameters	23
119	Table 9 – Operations: CIM_ComputerSystem.....	23
120	Table 10 – Operations: CIM_HostedService	24
121	Table 11 – Operations: CIM_IsSpare	24
122	Table 12 – Operations: CIM_ElementCapabilities	25
123	Table 13 – Operations: CIM_MemberOfCollection.....	25
124	Table 14 – Operations: CIM_ServiceAffectsElement	26
125	Table 15 – CIM Elements: Service Processor Profile.....	31
126	Table 16 – Class: CIM_ComputerSystem.....	31
127	Table 17 – Class: CIM_ElementCapabilities.....	32
128	Table 18 – Class: CIM_EnabledLogicalElementCapabilities.....	32
129	Table 19 – Class: CIM_HostedService	32
130	Table 20 – Class: CIM_IsSpare	33

131 Table 21 – Class: CIM_MemberOfCollection..... 33

132 Table 22 – Class: CIM_OwningCollectionElement 33

133 Table 23 – Class: CIM_RedundancySet..... 34

134 Table 24 – Class: CIM_RegisteredProfile..... 34

135 Table 25 – Class: CIM_ServiceAffectsElement 34

136 Table 26 – Class: CIM_TimeService 35

137 Table 27 – Class: CIM_ManagementController..... 35

138

140

Foreword

141 The *Service Processor Profile* (DSP1018) was prepared by the Physical Platform Profiles Working Group
142 and the Server Management Working Group of the DMTF.

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

145 **Acknowledgments**

146 The authors wish to acknowledge the following people.

147 **Editor:**

- 148 • Aaron Merkin – IBM
- 149 • Jeff Hilland - HP

150 **Contributors:**

- 151 • Jon Hass – Dell
- 152 • Khachatur Papanyan – Dell
- 153 • Enoch Suen – Dell
- 154 • Jeff Hilland – HP
- 155 • Christina Shaw – HP
- 156 • Aaron Merkin – IBM
- 157 • Perry Vincent – Intel
- 158 • John Leung – Intel
- 159 • Hemal Shah – Broadcom

160

161

Introduction

162 The information in this specification should be sufficient for a provider or consumer of this data to identify
163 unambiguously the classes, properties, methods, and values that shall be instantiated and manipulated to
164 represent and manage a service processor that is modeled using the DMTF Common Information Model
165 (CIM) core and extended model definitions.

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

168

Service Processor Profile

169 1 Scope

170 The *Service Processor Profile* is an autonomous profile for modeling service processors.

171 2 Normative References

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

175 2.1 Approved References

176 DMTF DSP0004, *CIM Infrastructure Specification 2.5*,
177 http://www.dmtf.org/standards/published_documents/DSP0004_2.5.pdf

178 DMTF DSP0200, *CIM Operations over HTTP 1.2*,
179 http://www.dmtf.org/standards/published_documents/DSP0200_1.2.pdf

180 DMTF DSP1001, *Management Profile Specification Usage Guide 1.0*,
181 http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf

182 DMTF DSP1004, *Base Server Profile 1.0*,
183 http://www.dmtf.org/standards/published_documents/DSP1004_1.0.pdf

184 DMTF DSP1005, *CLP Service Profile 1.0*,
185 http://www.dmtf.org/standards/published_documents/DSP1005_1.0.pdf

186 DMTF DSP1006, *SMASH Collections Profile 1.0*,
187 http://www.dmtf.org/standards/published_documents/DSP1006_1.0.pdf

188 DMTF DSP1008, *Modular System Profile 1.0*,
189 http://www.dmtf.org/standards/published_documents/DSP1008_1.0.pdf

190 DMTF DSP1009, *Sensors Profile 1.0*,
191 http://www.dmtf.org/standards/published_documents/DSP1009_1.0.pdf

192 DMTF DSP1010, *Record Log Profile 1.0*,
193 http://www.dmtf.org/standards/published_documents/DSP1010_1.0.pdf

194 DMTF DSP1011, *Physical Asset Profile 1.0*,
195 http://www.dmtf.org/standards/published_documents/DSP1011_1.0.pdf

196 DMTF DSP1012, *Boot Control Profile 1.0*,
197 http://www.dmtf.org/standards/published_documents/DSP1012_1.0.pdf

198 DMTF DSP1014, *Ethernet Port Profile 1.0*,
199 http://www.dmtf.org/standards/published_documents/DSP1014_1.0.pdf

200 DMTF DSP1016, *Telnet Service Profile 1.0*,
201 http://www.dmtf.org/standards/published_documents/DSP1016_1.0.pdf

202 DMTF DSP1017, *SSH Service Profile 1.0*,
203 http://www.dmtf.org/standards/published_documents/DSP1017_1.0.pdf

204 DMTF DSP1021, *Shared Device Management Profile 1.0*,
205 http://www.dmtf.org/standards/published_documents/DSP1021_1.0.pdf

206 DMTF DSP1023, *Software Inventory Profile 1.0*,
207 http://www.dmtf.org/standards/published_documents/DSP1023_1.0.pdf

208 DMTF DSP1024, *Text Console Redirection Profile 1.0*,
209 http://www.dmtf.org/standards/published_documents/DSP1024_1.0.pdf

210 DMTF DSP1025, *Software Update Profile 1.0*,
211 http://www.dmtf.org/standards/published_documents/DSP1025_1.0.pdf

212 DMTF DSP1027, *Power State Management Profile 1.0*,
213 http://www.dmtf.org/standards/published_documents/DSP1027_1.0.pdf

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

216 DMTF DSP1034, *Simple Identity Management Profile 1.0*,
217 http://www.dmtf.org/standards/published_documents/DSP1034_1.0.pdf

218 DMTF DSP1036, *IP Interface Profile 1.0*,
219 http://www.dmtf.org/standards/published_documents/DSP1036_1.0.pdf

220 DMTF DSP1037, *DHCP Client Profile 1.0*,
221 http://www.dmtf.org/standards/published_documents/DSP1037_1.0.pdf

222 DMTF DSP1038, *DNS Client Profile 1.0*,
223 http://www.dmtf.org/standards/published_documents/DSP1038_1.0.pdf

224 DMTF DSP1039, *Role Based Authorization Profile 1.0*,
225 http://www.dmtf.org/standards/published_documents/DSP1039_1.0.pdf

226 DMTF DSP1075, *PCI Device Profile 1.0*,
227 http://www.dmtf.org/standards/published_documents/DSP1075_1.0.pdf

228 **2.2 Other References**

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

231 **3 Terms and Definitions**

232 For the purposes of this document, the terms and definitions in [DSP1033](#) and [DSP1001](#) and the following
233 apply.

234 **3.1**

235 **can**

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

237 **3.2**

238 **cannot**

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

240 **3.3**

241 **conditional**

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

- 244 **3.4**
245 **mandatory**
246 indicates requirements to be followed strictly to conform to the document and from which no deviation is
247 permitted
- 248 **3.5**
249 **may**
250 indicates a course of action permissible within the limits of the document
- 251 **3.6**
252 **need not**
253 indicates a course of action permissible within the limits of the document
- 254 **3.7**
255 **optional**
256 indicates a course of action permissible within the limits of the document
- 257 **3.8**
258 **referencing profile**
259 indicates a profile that owns the definition of this class and can include a reference to this profile in its
260 "Referenced Profiles" table
- 261 **3.9**
262 **shall**
263 indicates requirements to be followed strictly to conform to the document and from which no deviation is
264 permitted
- 265 **3.10**
266 **shall not**
267 indicates requirements to be followed strictly to conform to the document and from which no deviation is
268 permitted
- 269 **3.11**
270 **should**
271 indicates that among several possibilities, one is recommended as particularly suitable, without
272 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 273 **3.12**
274 **should not**
275 indicates that a certain possibility or course of action is deprecated but not prohibited
- 276 **3.13**
277 **unspecified**
278 indicates that this profile does not define any constraints for the referenced CIM element or operation
- 279 **3.14**
280 **service processor**
281 a specialized device dedicated to management
- 282 **3.15**
283 **standby service processor**
284 an instance of CIM_ComputerSystem that represents a standby service processor of a redundancy set

285 4 Symbols and Abbreviated Terms

286 None.

287 5 Synopsis

288 **Profile Name:** Service Processor

289 **Version:** 1.1.0

290 **Organization:** DMTF

291 **CIM Schema Version:** 2.20

292 **Central Class:** CIM_ComputerSystem

293 **Scoping Class:** CIM_ComputerSystem

294 Table 1 identifies profiles on which this profile has a dependency.

295

Table 1 – Referenced Profiles

Profile Name	Organization	Version	Relationship	Behavior
Simple Identity Management	DMTF	1.0	Optional	See 7.3.
Boot Control	DMTF	1.0	Optional	See 7.5.
CLP Service	DMTF	1.0	Optional	See 7.6.
DHCP Client	DMTF	1.0	Optional	See 7.7.
DNS Client	DMTF	1.0	Optional	See 7.8.
Ethernet Port	DMTF	1.0	Optional	See 7.9.
Software Inventory	DMTF	1.0	Optional	See 7.10.
Software Update	DMTF	1.0	Optional	See 7.11.
IP Interface	DMTF	1.0	Optional	See 7.12.
Physical Asset	DMTF	1.0	Optional	See 7.13.
Profile Registration	DMTF	1.0	Mandatory	None
Record Log	DMTF	1.0	Optional	See 7.14.
Role Based Authorization	DMTF	1.0	Optional	See 7.3.
Sensors	DMTF	1.0	Optional	See 7.15.
Power State Management	DMTF	1.0	Optional	See 7.16.
Shared Device Management	DMTF	1.0	Optional	See 7.17.
SMASH Collections	DMTF	1.0	Optional	See 7.18.
SSH Service	DMTF	1.0	Optional	See 7.19.
Telnet Service	DMTF	1.0	Optional	See 7.20.
Text Console Redirection	DMTF	1.0	Optional	See 7.21.
PCI Device	DMTF	1.0	Optional	See 7.22.

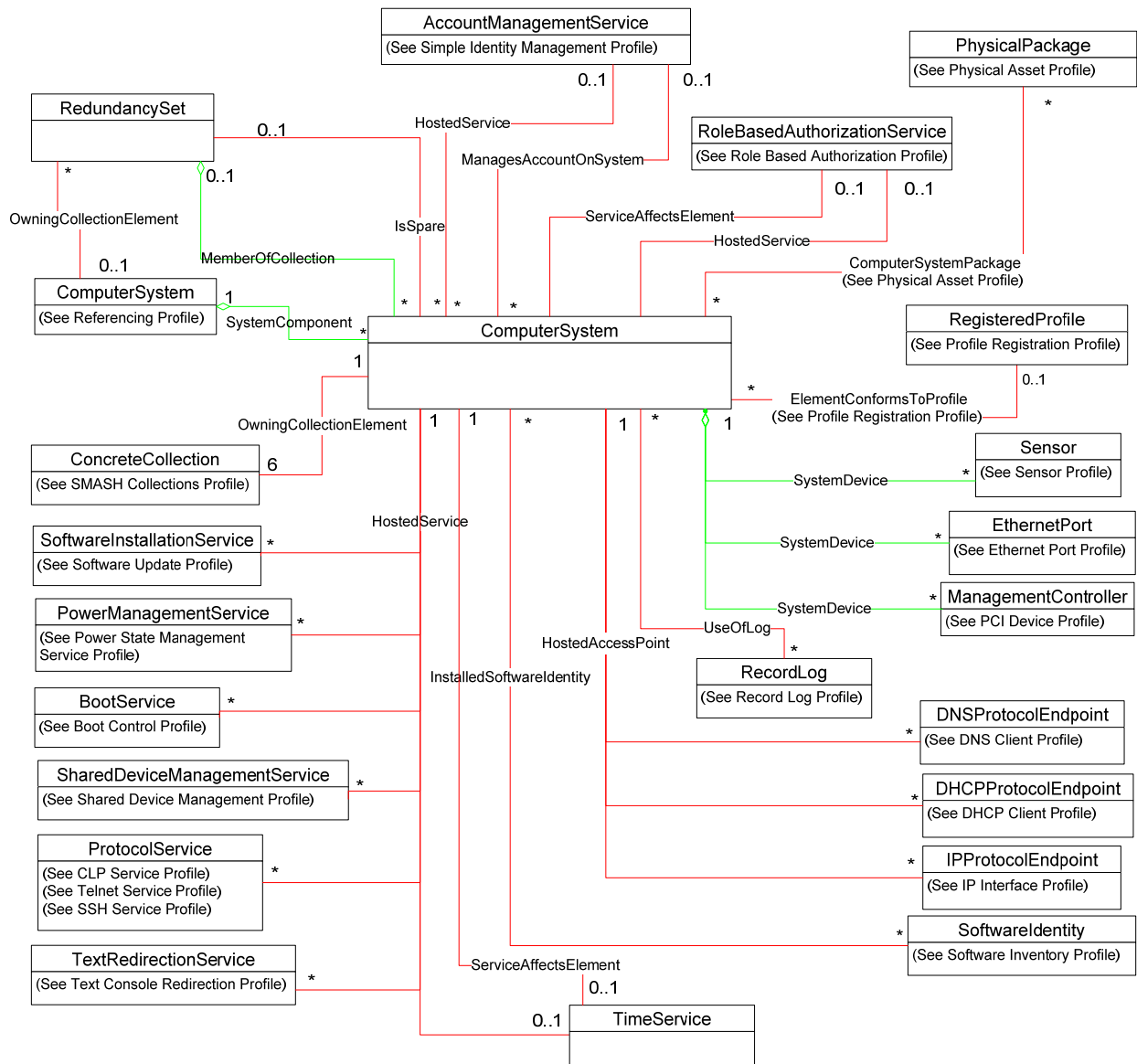
296 **6 Description**

297 The *Service Processor Profile* describes the management and configuration of a service processor for a
298 computer system. The computer system may be contained in a single chassis or comprise a more
299 complex modular system with multiple chassis or a blade system. This description includes modeling
300 redundant service processors.

301 Some examples of the service processors are:

- 302 • management processor (MP)
- 303 • service processor (SP)
- 304 • baseboard management controller (BMC)
- 305 • chassis manager

306 Figure 1 represents the class schema for the *Service Processor Profile*. For simplicity, the prefix CIM_
307 has been removed from the names of the classes.



308

309

Figure 1 – Service Processor Profile: Class Diagram

310 7 Implementation

311 This section details the requirements related to the arrangement of instances and their properties for
 312 implementations of this profile. All required methods and operations are described in clause 8. Required
 313 CIM elements are described in clause 10.

314 7.1 Representing a Service Processor

315 A service processor shall be represented with an instance of CIM_ComputerSystem.

316 7.1.1 CIM_ComputerSystem.EnabledState

317 Table 2 describes the mapping between the values of the CIM_ComputerSystem.EnabledState property
 318 and the corresponding description of the state of the service processor. The EnabledState property shall
 319 match the values that are specified in Table 2. When the RequestStateChange() method executes but
 320 does not complete successfully, and the service processor is in an indeterminate state, the EnabledState
 321 property shall have value of 5 (Not Applicable). The value of the EnabledState property may also change
 322 as a result of change to the service processor's enabled state by non-CIM implementation.

323 **Table 2 – CIM_ComputerSystem.EnabledState Value Description**

Value	Description	Extended Description
2	Enabled	The service processor shall be enabled.
3	Disabled	The service processor shall be disabled.
5	Not Applicable	The service processor state is indeterminate, or service processor state management is not supported.
6	Enabled but Offline	The service processor shall be enabled but inactive (used in redundant configuration; see 7.2.4).

324 7.1.2 Service Processor State Management Is Supported — Conditional

325 Support for managing the state of the service processor is optional behavior. This section describes the
 326 CIM elements and behaviors that shall be implemented when this behavior is supported.

327 7.1.2.1 CIM_EnabledLogicalElementCapabilities

328 When state management is supported, exactly one instance of CIM_EnabledLogicalElementCapabilities
 329 shall be associated with the CIM_ComputerSystem instance that represents a service processor through
 330 an instance of CIM_ElementCapabilities.

331 7.1.2.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported

332 The RequestedStatesSupported property may contain zero or more of the following values: 2 (Enabled),
 333 3 (Disabled), 6 (Offline), or 11 (Reset).

334 7.1.2.2 CIM_ComputerSystem.RequestedState

335 When the CIM_ComputerSystem.RequestStateChange() method is successfully invoked, the value of the
 336 RequestedState property shall be the value of the RequestedState parameter. If the method is not
 337 successfully invoked, the value of the RequestedState property is indeterminate.

338 The CIM_ComputerSystem.RequestedState property shall have one of the values specified in the
 339 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property or a value of 5 (No
 340 Change).

341 7.1.2.3 CIM_ComputerSystem.EnabledState

342 When the RequestedState parameter has a value of 2 (Enabled) or 3 (Disabled) and the
 343 CIM_ComputerSystem.RequestStateChange() method completes successfully, the value of the
 344 EnabledState property shall equal the value of the CIM_ComputerSystem.RequestedState property.

345 If the method does not complete successfully, the value of the EnabledState property is indeterminate.

346 7.1.3 Service Processor State Management Is Not Supported

347 This section describes the CIM elements and behaviors that shall be implemented when management of
 348 the service processor state is not supported.

349 **7.1.3.1 CIM_EnabledLogicalElementCapabilities**

350 When state management is not supported, exactly one instance of
351 CIM_EnabledLogicalElementCapabilities may be associated with the CIM_ComputerSystem instance that
352 represents a service processor through an instance of CIM_ElementCapabilities.

353 **7.1.3.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported**

354 The CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall not contain any
355 values.

356 **7.1.3.2 CIM_ComputerSystem.RequestedState**

357 The RequestedState property shall have the value 12 (Not Applicable).

358 **7.1.4 Modifying ElementName Is Supported — Conditional**

359 The CIM_ComputerSystem.ElementName property may support being modified by the ModifyInstance
360 operation. See 8.5.1. This behavior is conditional. This section describes the CIM elements and behavior
361 requirements when an implementation supports client modification of the
362 CIM_ComputerSystem.ElementName property.

363 **7.1.4.1 CIM_EnabledLogicalElementCapabilities**

364 An instance of CIM_EnabledLogicalElementCapabilities shall be associated with the
365 CIM_ComputerSystem instance through an instance of CIM_ElementCapabilities.

366 **7.1.4.1.1 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported**

367 The ElementNameEditSupported property shall have a value of TRUE.

368 **7.1.4.1.2 CIM_EnabledLogicalElement.MaxElementNameLen**

369 The MaxElementNameLen property shall be implemented.

370 **7.1.5 Modifying ElementName Is Not Supported**

371 This section describes the CIM elements and behaviors that shall be implemented when the
372 CIM_ComputerSystem.ElementName does not support being modified by the ModifyInstance operation.

373 **7.1.5.1 CIM_EnabledLogicalElementCapabilities**

374 An instance of CIM_EnabledLogicalElementCapabilities may be associated with the
375 CIM_ComputerSystem instance through an instance of CIM_ElementCapabilities.

376 **7.1.5.1.1 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported**

377 The ElementNameEditSupported shall have a value of FALSE.

378 **7.1.5.1.2 CIM_EnabledLogicalElement.MaxElementNameLen**

379 The MaxElementNameLen property may be implemented. The MaxElementNameLen property is
380 irrelevant in this context.

381 **7.1.6 Representing the Physical Packaging (Optional)**

382 Support for representing the physical packaging of the service processor is optional. The physical
383 packaging may be modeled using one or more instances of CIM_PhysicalElement in accordance with the
384 [Physical Asset Profile](#).

385 7.2 Modeling Service Processor Redundancy (Optional)

386 Modeling of service processor redundancy is optional. When service processor redundancy is supported,
387 the requirements in this section apply.

388 At least one instance of CIM_RedundancySet shall exist.

389 7.2.1 Relationship between Redundancy Set and Redundant Service Processors

390 Each CIM_ComputerSystem instance that represents a service processor participating in the redundancy
391 shall be associated with the CIM_RedundancySet instance through the CIM_MemberOfCollection
392 association. Each instance of CIM_ComputerSystem that is associated with the CIM_RedundancySet
393 instance through the CIM_MemberOfCollection association shall be associated with the same instance of
394 CIM_ComputerSystem through the CIM_SystemComponent association where the value of the
395 CIM_SystemComponent.PartComponent property is the instance of CIM_ComputerSystem that is
396 associated with the CIM_RedundancySet.

397 7.2.2 Relationship between Redundancy Set and Containing System

398 When the CIM_ComputerSystem instance that represents a service processor is associated with another
399 CIM_ComputerSystem instance through the CIM_SystemComponent association where the value of the
400 CIM_SystemComponentPartComponent property is the CIM_ComputerSystem instance that represents
401 the service processor, the CIM_RedundancySet instance shall be associated with the
402 CIM_ComputerSystem instance that is the value of the CIM_SystemComponent.GroupComponent
403 property through the CIM_OwningCollectionElement association.

404 7.2.3 Active / Active Redundancy

405 When the CIM_RedundancySet.TypeOfSet property contains a value of 3 (Load Balanced) or 2 (N+1),
406 the CIM_ComputerSystem instances that are associated the CIM_RedundancySet instance shall comply
407 with the following requirements:

- 408 • The CIM_ComputerSystem instances shall not be associated with the CIM_RedundancySet
409 instance through the CIM_IsSpare association.
- 410 • For each instance of CIM_ComputerSystem, the CIM_ComputerSystem.EnabledState property
411 shall not have the value 6 (Enabled but Offline).

412 7.2.4 Active / Standby Redundancy

413 When the CIM_RedundancySet.TypeOfSet property contains a value of 4 (Sparing) or 5 (Limited
414 Sparing), one or more standby service processor s may exist. Each standby service processor shall be
415 associated to the CIM_RedundancySet instance through the CIM_IsSpare association.

416 Each standby service processor shall comply with one of the following requirements:

- 417 • When the CIM_ComputerSystem.EnabledState property has the value 6 (Enabled but Offline),
418 the SpareStatus property of the referencing CIM_IsSpare instance shall have the value 2 (Hot
419 Standby).
- 420 • When the CIM_ComputerSystem.EnabledState property has the value 3 (Disabled), the
421 SpareStatus property of the referencing CIM_IsSpare instance shall have the value 3 (Cold
422 Standby).
- 423 • When the CIM_ComputerSystem.EnabledState property has a value other than 3 (Disabled) or
424 6 (Enabled but Offline), the SpareStatus property of the referencing CIM_IsSpare instance shall
425 have the value 0 (Unknown).

426 7.3 Managing Service Processor Time (Optional)

427 A service processor can maintain an internal clock. This internal clock provides the service processor with
428 the current time (for example, to provide time stamps for log entries). Management of the current time of
429 the service processor may be supported. This behavior is optional. When management of the current time
430 of the service processor is supported, the requirements specified in this section shall be met.

431 An instance of CIM_TimeService shall be associated with the Central Instance through the
432 CIM_HostedService association. The instance of CIM_TimeService shall also be associated with the
433 Central Instance through the CIM_ServiceAffectsElement association.

434 7.4 User Account Management (Optional)

435 The [Simple Identity Management Profile](#) and the [Role Based Authorization Profile](#) may be implemented to
436 model user access to the service processor. When the [Simple Identity Management Profile](#) is
437 implemented, an instance of CIM_AccountManagementService shall be associated with the Central
438 Instance through the CIM_HostedService association. When the [Role Based Authorization Profile](#) is
439 implemented, an instance of CIM_RoleBasedAuthorizationService shall be associated with the Central
440 Instance through the CIM_HostedService association.

441 7.5 Boot Control Profile (Optional)

442 The [Boot Control Profile](#) may be implemented to model the ability of the service processor to manage its
443 own boot configuration or that of the systems it managed. If the [Boot Control Profile](#) is implemented, an
444 instance of CIM_BootService shall be associated with the Central Instance through the
445 CIM_HostedService association.

446 7.6 CLP Service Profile (Optional)

447 The [CLP Service Profile](#) may be implemented to model a CLP service hosted on the service processor.
448 When the [CLP Service Profile](#) is implemented, at least one instance of CIM_ProtocolService shall be
449 associated with the Central Instance through an instance of CIM_HostedService.

450 7.7 DHCP Client Profile (Optional)

451 The [DHCP Client Profile](#) may be implemented to model the DHCP client of a service processor. When the
452 [DHCP Client Profile](#) is implemented, at least one instance of CIM_DHCPProtocolEndpoint shall be
453 associated with the Central Instance through an instance of CIM_HostedAccessPoint.

454 7.8 DNS Client Profile (Optional)

455 The [DNS Client Profile](#) may be implemented to model the DNS client of a service processor. When the
456 [DNS Client Profile](#) is implemented, at least one instance of CIM_DNSProtocolEndpoint shall be
457 associated with the Central Instance through an instance of CIM_HostedAccessPoint.

458 7.9 Ethernet Port Profile (Optional)

459 The [Ethernet Port Profile](#) may be implemented to model an Ethernet interface of a service processor.
460 When the [Ethernet Port Profile](#) is implemented, at least one instance of CIM_EthernetPort shall be
461 associated with the Central Instance through an instance of CIM_SystemDevice.

462 7.10 Software Inventory Profile (Optional)

463 The [Software Inventory Profile](#) may be implemented to model the software version information of the
464 service processor. When the [Software Inventory Profile](#) is implemented, at least one instance of
465 CIM_SoftwareIdentity shall be associated with the Central Instance of this profile through an instance of
466 CIM_InstalledSoftwareIdentity.

467 **7.11 Software Update Profile (Optional)**

468 The [Software Update Profile](#) may be implemented to model the ability of the service processor to update
469 software installed on one or more components of managed systems, including the service processor
470 itself. When the [Software Update Profile](#) is implemented, an instance of CIM_SoftwareInstallationService
471 shall be associated with the Central Instance through and instance of CIM_HostedService.

472 **7.12 IP Interface Profile (Optional)**

473 The [IP Interface Profile](#) may be implemented to model the IP interface of a service processor. When the
474 [IP Interface Profile](#) is implemented, at least one instance of CIM_IPProtocolEndpoint shall be associated
475 with the Central Instance through an instance of CIM_HostedAccessPoint.

476 **7.13 Physical Asset Profile (Optional)**

477 The [Physical Asset Profile](#) may be implemented to model the physical package and physical asset
478 information of a service processor. When the [Physical Asset Profile](#) is implemented, at least one instance
479 of CIM_PhysicalPackage shall be associated with the Central Instance through an instance of
480 CIM_ComputerSystemPackage.

481 **7.14 Record Log Profile (Optional)**

482 The [Record Log Profile](#) may be implemented to model one or more logs of the service processor. When
483 the [Record Log Profile](#) is implemented, an instance of CIM_RecordLog shall be associated with Central
484 Instance through an instance of CIM_UseOfLog.

485 **7.15 Sensors Profile (Optional)**

486 The [Sensors Profile](#) may be implemented to model the sensors of the service processor. When the
487 [Sensors Profile](#) is implemented, at least one instance of CIM_Sensor or CIM_NumericSensor shall be
488 associated with the Central Instance through an instance of CIM_SystemDevice.

489 **7.16 Power State Management Profile (Optional)**

490 The [Power State Management Profile](#) may be implemented to model the ability of the service processor
491 to perform power control operations for the managed system or the service processor itself. When the
492 [Power State Management Profile](#) is implemented, an instance of CIM_PowerManagementService shall
493 be associated with the Central Instance through an instance of CIM_HostedService.

494 **7.17 Shared Device Management Profile (Optional)**

495 The [Shared Device Management Profile](#) may be implemented to model the ability of the service
496 processor to control shared devices of a modular system. When the [Shared Device Management Profile](#)
497 is implemented, an instance of CIM_SharedDeviceManagementService shall be associated with the
498 Central Instance through an instance of CIM_HostedService.

499 **7.18 SMASH Collections Profile (Optional)**

500 The [SMASH Collections Profile](#) may be implemented. When the [SMASH Collections Profile](#) is
501 implemented, each instance of CIM_ConcreteCollection that is defined by the [SMASH Collections Profile](#)
502 shall be associated with the Central Instance through an instance of CIM_OwningCollectionElement.

503 **7.19 SSH Service Profile (Optional)**

504 The [SSH Service Profile](#) may be implemented to model an SSH service hosted on the service processor.
505 When the [SSH Service Profile](#) is implemented, at least one instance of CIM_ProtocolService shall be

506 associated with the Central Instance through an instance of CIM_HostedService where the
507 CIM_ProtocolService.Protocol property has the value 2 (SSH).

508 **7.20 Telnet Service Profile (Optional)**

509 The [Telnet Service Profile](#) may be implemented to model a Telnet service hosted on the service
510 processor. When the [Telnet Service Profile](#) is implemented, at one instance of CIM_ProtocolService shall
511 be associated with the Central Instance through an instance of CIM_HostedService where the
512 CIM_ProtocolService.Protocol property has the value 3 (Telnet).

513 **7.21 Text Console Redirection Profile (Optional)**

514 The [Text Console Redirection Profile](#) may be implemented to model the ability of the service processor to
515 provide text console redirection for managed systems. When the [Text Console Redirection Profile](#) is
516 implemented, at least one instance of CIM_TextRedirectionService shall be associated with the Central
517 Instance through an instance of CIM_HostedService.

518 **7.22 PCI Device Profile (Optional)**

519 The [PCI Device Profile](#) may be implemented to model the ability of the service processor to provide PCI
520 configuration information for managed systems. When the [PCI Device Profile](#) is implemented and the
521 ServiceProcessor is modeled as a PCI device, at least one instance of CIM_ManagementController shall
522 be associated with the Central Instance of this profile through an instance of CIM_SystemDevice and the
523 CIM_ManagementController shall be associated with at least one instance of CIM_PCIDevice through an
524 instance of CIM_ConcretelIdentity.

525 **8 Methods**

526 This section details the requirements for supporting intrinsic operations and extrinsic methods for the CIM
527 elements defined by this profile.

528 **8.1 Method: CIM_ComputerSystem.RequestStateChange()**

529 Invocation of the CIM_ComputerSystem.RequestStateChange() method changes the element's state to
530 the value specified in the RequestedState parameter.

531 Return values for the RequestStateChange() method are specified in Table 3. Parameters for the
532 RequestStateChange() method are specified in Table 4.

533 The RequestStateChange() method shall be implemented and shall not return a value of 1 (Not
534 Supported) when state management of the service processor is supported (see 7.1.2).

535 When the RequestedState parameter has a value of 6 (Offline) and the CIM_ComputerSystem instance is
536 not a standby service processor, the RequestStateChange() method shall return a value of 2 (Error
537 Occurred).

538 Invoking the RequestStateChange() method multiple times could result in earlier requests being
539 overwritten or lost.

540 No standard messages are defined for this method.

541 **Table 3 – CIM_ComputerSystem.RequestStateChange() Method: Return Code Values**

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred.
4096	Job started.

542 **Table 4 – CIM_ComputerSystem.RequestStateChange() Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	RequestedState	uint16	2 (Enabled) 3 (Disabled), see 8.1.1 6 (Offline), see 8.1.1 11 (Reset)
OUT	Job	CIM_ConcreteJob REF	Returned if job started
IN, REQ	TimeoutPeriod	Datetime	Client specified maximum amount of time the transition to a new state is supposed to take: 0 or NULL – No time requirements <interval> – Maximum time allowed

543 **8.1.1 RequestStateChange() for the Standby Service Processor**

544 After the successful execution of the RequestStateChange() method on the standby service processor
545 with the RequestedState parameter set to 6 (Offline), the SpareStatus property of the referenced
546 CIM_IsSpare association shall have a value of 2 (Hot Standby).

547 After the successful execution of the RequestStateChange() method on the standby service processor
548 with the RequestedState parameter set to 3 (Disabled), the SpareStatus property of the referenced
549 CIM_IsSpare association shall have value of 3 (Cold Standby).

550 **8.2 Method: CIM_RedundancySet.Failover()**

551 The CIM_RedundancySet.Failover() method forces a failover from one member of a
552 CIM_RedundancySet collection to another. After the successful execution of the method, the service
553 processor that is represented by the CIM_ComputerSystem instance referenced by the FailoverFrom
554 parameter becomes inactive. The service processor that is represented by CIM_ComputerSystem
555 instance referenced by the FailoverTo parameter takes over as the active service processor.

556 The Failover() method may be supported if the FailoverSupported property of at least one instance of
557 CIM_IsSpare that references the CIM_RedundancySet instance has a value of 3 (Manual) or 4 (Both
558 Manual and Automatic).

559 The Failover() method shall not be supported if the FailoverSupported property of every instance of
560 CIM_IsSpare that references the CIM_RedundancySet instance has a value of 2 (Automatic).

561 The execution of the Failover() method shall return a value of 2 (Error Occurred) under the following
562 circumstances:

- 563 • The CIM_ComputerSystem instance that is referenced by the FailoverTo parameter is not a
564 standby service processor.

- 565 • The CIM_ComputerSystem instance that is referenced by the FailoverFrom parameter is not
566 associated with the CIM_RedundancySet instance only through the CIM_MemberOfCollection
567 association.

568 After the successful execution of the Failover() method, the following events occur:

- 569 • The CIM_ComputerSystem that is referenced by the FailoverTo parameter shall take over as the
570 active service processor.
- 571 • The CIM_ComputerSystem instance that is referenced by the FailoverTo parameter shall be
572 associated with the CIM_RedundancySet instance only through the CIM_MemberOfCollection
573 association.
- 574 • The CIM_ComputerSystem instance that is referenced by the FailoverFrom parameter shall
575 become a standby service processor. This instance will conform to the requirements for a
576 standby service processor specified in 7.2.4.
- 577 • When management of the service processor state is supported, the CIM_ComputerSystem
578 instance that is referenced by the FailoverFrom parameter shall not have an EnabledState
579 property value of 2 (Enabled) but may have a value of 6 (Enabled but Offline).

580 Return code values for the CIM_RedundancySet.Failover() method are specified in Table 5. Parameters
581 for the CIM_RedundancySet.Failover() method are specified in Table 6. No standard messages are
582 defined for this method.

583 **Table 5 – CIM_RedundancySet.Failover() Method: Return Code Values**

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred.

584 **Table 6 – CIM_RedundancySet.Failover() Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	FailoverFrom	CIM_ManagedElement REF	The redundant element that will become inactive
IN, REQ	FailoverTo	CIM_ManagedElement REF	The redundant element that will become active and take over the inactivated element

585 **8.3 Method: CIM_TimeService.ManageTime()**

586 The CIM_TimeService.ManageTime() method is used to query or modify the service processor time.
587 When the GetRequest parameter has a value of TRUE, the TimeData parameter shall be ignored. If the
588 GetRequest parameter is not specified, the method shall return a value of 2 (Error Occurred). When the
589 ManagedElement parameter is not a reference to the Central Instance, the method shall return a value of
590 2 (Error Occurred).

591 Detailed requirements of the CIM_TimeService() method are specified in Table 7 and Table 8. No
592 standard messages are defined for this method.

593 **Table 7 – CIM_TimeService.ManageTime() Method: Return Code Values**

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred.

594 **Table 8 – CIM_TimeService.ManageTime() Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	GetRequest	Boolean	Indicates whether the request is to get the time (TRUE) or set the time (FALSE) for the specified element
IN / OUT	TimeData	datetime	On input, this is the desired value for the service processor time. On output, this is the service processor time.
IN	ManagedElement	CIM_Managed Element	Reference to the Central Instance

595 **8.4 Profile Conventions for Operations**

596 For each profile class (including associations), the implementation requirements for operations, including
 597 those in the following default list, are specified in class-specific subclauses of this clause.

598 The default list of operations is as follows:

- 599 • GetInstance
- 600 • Associators
- 601 • AssociatorNames
- 602 • References
- 603 • ReferenceNames
- 604 • EnumerateInstances
- 605 • EnumerateInstanceNames

606 **8.5 CIM_ComputerSystem**

607 Table 9 lists implementation requirements for operations. If implemented, these operations shall be
 608 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 9, all operations in
 609 the default list in 8.4 shall be implemented as defined in [DSP0200](#).

610 NOTE: Related profiles may define additional requirements on operations for the profile class.

611 **Table 9 – Operations: CIM_ComputerSystem**

Operation	Requirement	Messages
ModifyInstance	Optional. See 8.5.1.	None

612 **8.5.1 CIM_ComputerSystem — ModifyInstance**

613 This section details the requirements for the ModifyInstance operation applied to an instance of
 614 CIM_ComputerSystem. The ModifyInstance operation may be supported.

615 The ModifyInstance operation shall be supported and the CIM_ComputerSystem.ElementName property
 616 shall be modifiable when the ElementNameEditSupported property of the
 617 CIM_EnabledLogicalElementCapabilities instance that is associated with the CIM_ComputerSystem
 618 instance has a value of TRUE. See 8.5.1.1.

619 **8.5.1.1 CIM_ComputerSystem.ElementName**

620 When the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance
 621 that is associated with the CIM_ComputerSystem instance has a value of TRUE, the implementation shall
 622 allow the ModifyInstance operation to change the value of the ElementName property of the
 623 CIM_ComputerSystem instance. The ModifyInstance operation shall enforce the length restriction
 624 specified in the MaxElementNameLen property of the CIM_EnabledLogicalElementCapabilities instance.

625 When the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance
 626 has a value of FALSE, the implementation shall not allow the ModifyInstance operation to change the
 627 value of the ElementName property of the CIM_ComputerSystem instance.

628 **8.6 CIM_HostedService**

629 Table 10 lists implementation requirements for operations. If implemented, these operations shall be
 630 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 10, all operations
 631 in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

632 NOTE: Related profiles may define additional requirements on operations for the profile class.

633 **Table 10 – Operations: CIM_HostedService**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

634 **8.7 CIM_IsSpare**

635 Table 11 lists implementation requirements for operations. If implemented, these operations shall be
 636 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 11, all operations
 637 in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

638 NOTE: Related profiles may define additional requirements on operations for the profile class.

639 **Table 11 – Operations: CIM_IsSpare**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

640 8.8 CIM_ElementCapabilities

641 Table 12 lists implementation requirements for operations. If implemented, these operations shall be
 642 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 12, all operations
 643 in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

644 NOTE: Related profiles may define additional requirements on operations for the profile class.

645 **Table 12 – Operations: CIM_ElementCapabilities**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

646 8.9 CIM_EnabledLogicalElementCapabilities

647 All operations in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

648 NOTE: Related profiles may define additional requirements on operations for the profile class.

649 8.10 CIM_MemberOfCollection

650 Table 13 lists implementation requirements for operations. If implemented, these operations shall be
 651 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 13, all operations
 652 in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

653 NOTE: Related profiles may define additional requirements on operations for the profile class.

654 **Table 13 – Operations: CIM_MemberOfCollection**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

655 8.11 CIM_RedundancySet

656 All operations in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

657 NOTE: Related profiles may define additional requirements on operations for the profile class.

658 8.12 CIM_TimeService

659 All operations in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

660 NOTE: Related profiles may define additional requirements on operations for the profile class.

661 **8.13 CIM_ServiceAffectsElement**

662 Table 14 lists implementation requirements for operations. If implemented, these operations shall be
 663 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 14, all operations
 664 in the default list in 8.4 shall be implemented as defined in [DSP0200](#).

665 NOTE: Related profiles may define additional requirements on operations for the profile class.

666 **Table 14 – Operations: CIM_ServiceAffectsElement**

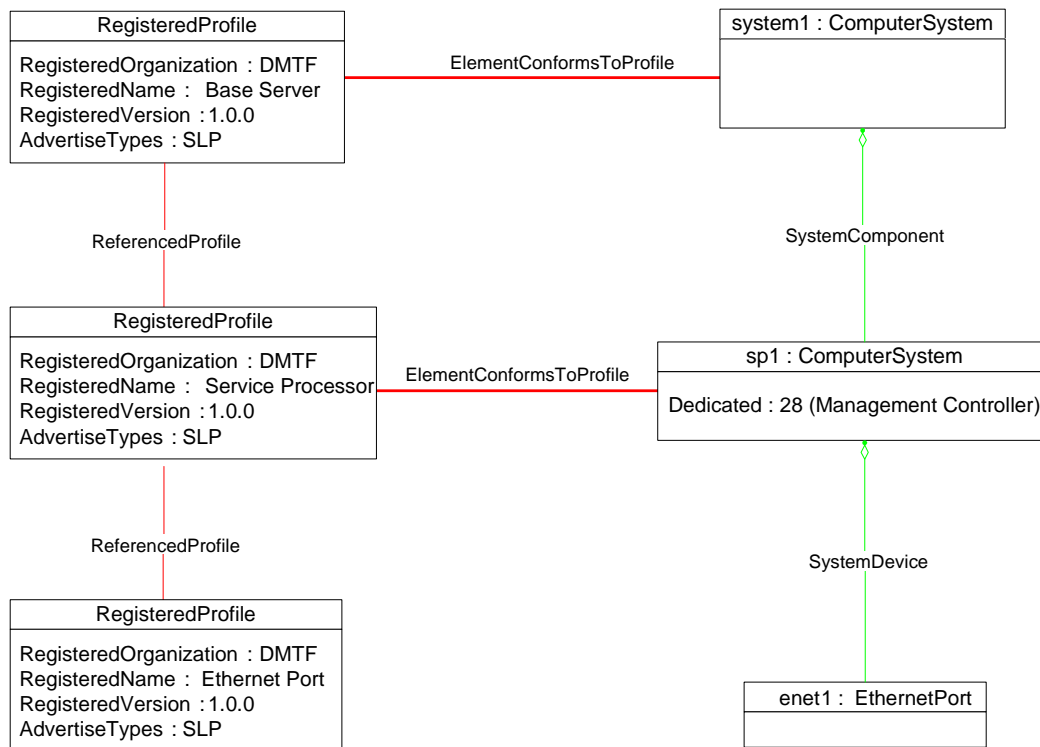
Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

667 **9 Use Cases**

668 This section contains object diagrams and use cases for the *Service Processor Profile*.

669 **9.1 Object Diagrams**

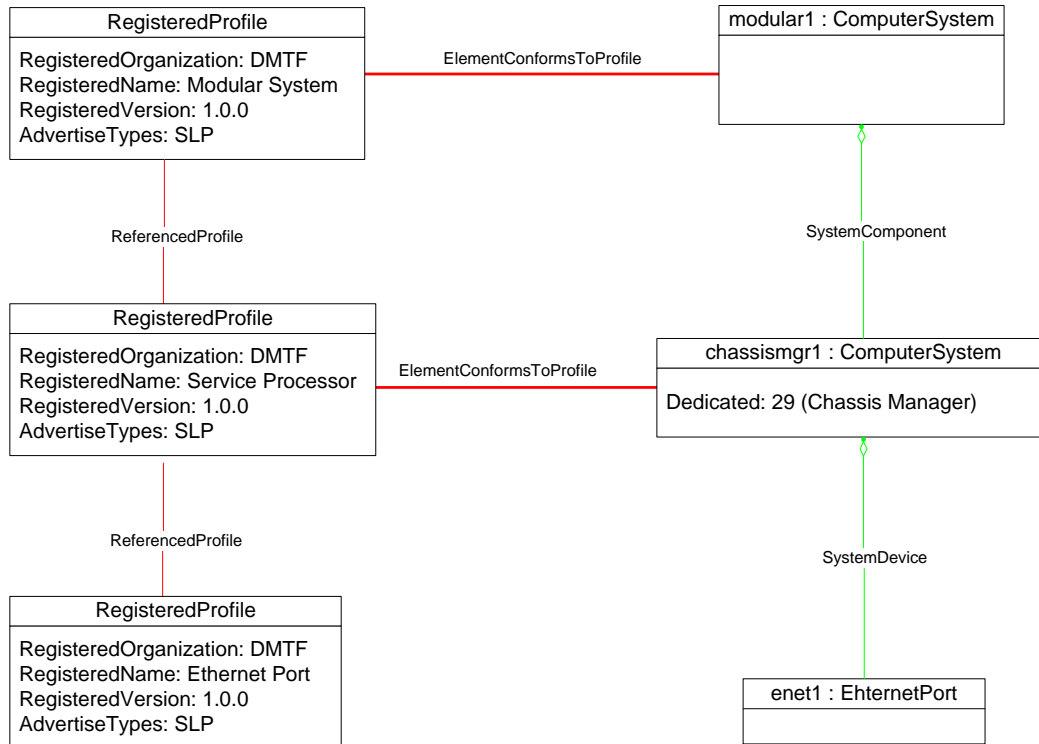
670 Figure 2 depicts an implementation of a service processor dedicated to a single computer system. Notice
 671 that the dedicated property of sp1 is 29 (Management Controller) and the managed computer system,
 672 system1 implements the [Base Server Profile](#). Figure 3 depicts an implementation of a Modular System
 673 with a chassis manager. Notice that the dedicated property of chassismgr1 is 29 (Chassis Manager) and
 674 that the manage system implements the [Modular System Profile](#).



675

676

Figure 2 – Base Server



677

678

Figure 3 – Modular System

679

Figure 4 is an object diagram showing redundant service processors installed in a modular system.

680

chassismgr1 is the active service processor. chassismgr2 is the backup service processor. This is

681

indicated by the values of the EnabledState and RequestedState properties of the two instances and by

682

the CIM_IsSpare association between the CIM_RedundancySet instance and chassismgr2.

683

In the illustrated system, a single configuration exists for the service processors. All functionality, including

684

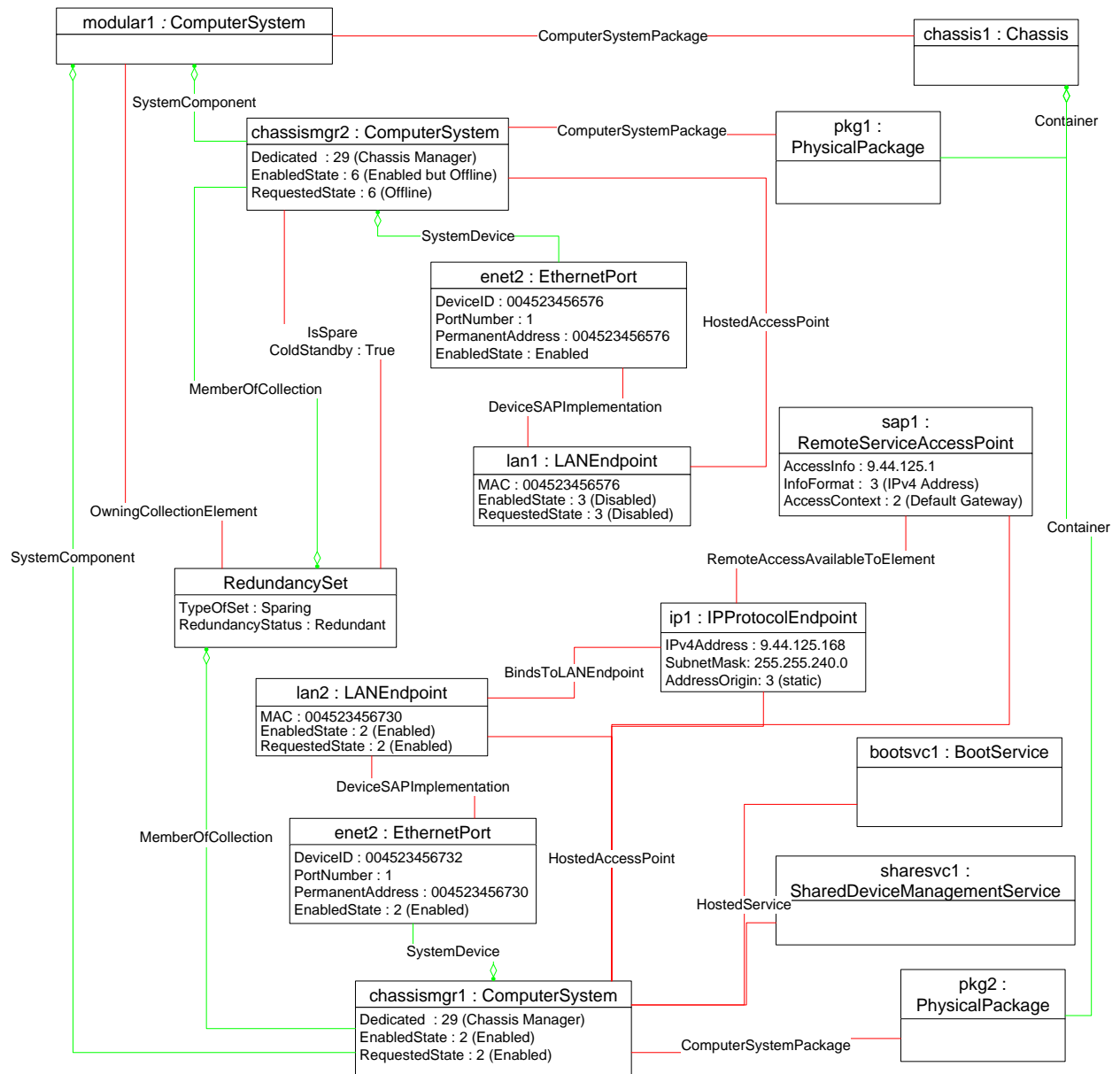
management interfaces, is hosted on and accessed at the active service processor. This is indicated by

685

the active IP interface (ip1) bound to the Ethernet interface (enet2) of chassismgr1 and by the services

686

(bootsvc1 and sharesvc1) associated through CIM_HostedService with chassismgr1.

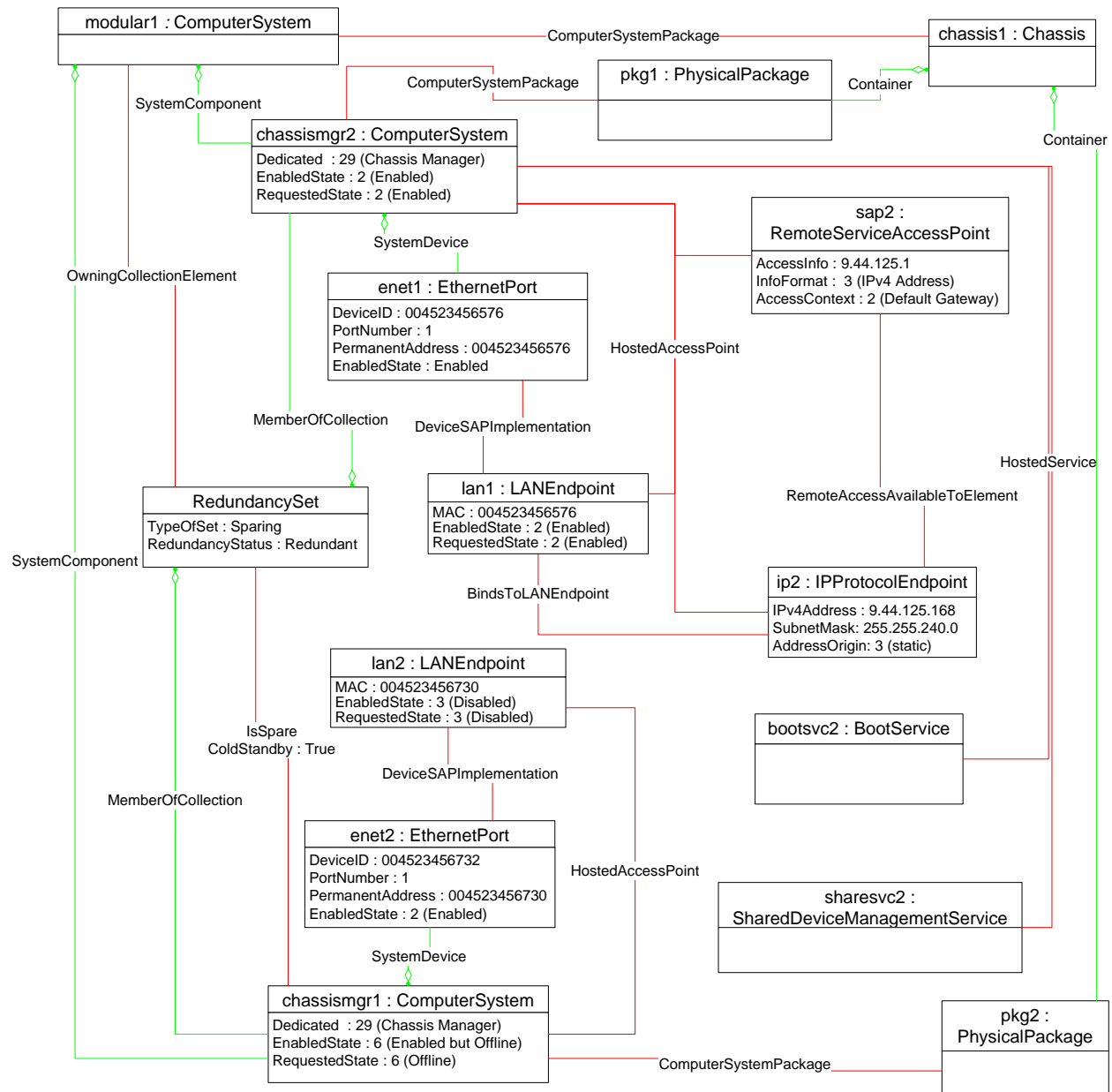


687

688

Figure 4 – Service Processors before Failover

689 Figure 5 shows the same system after a client has initiated a failover from chassismgr1 to chassismgr2.
 690 chassismgr2 is now the active service processor and chassismgr1 is the backup. The management
 691 functionality supported by the service processors of the system is now hosted on chassismgr2. Note that
 692 due to propagated key properties in the classes CIM_BootService,
 693 CIM_SharedDeviceManagementService, CIM_IPProtocolEndpoint, and
 694 CIM_RemoteServiceAccessPoint, new instances with identical values for the relevant properties are
 695 created and associated with chassismgr2 rather than merely changing the associated
 696 CIM_ComputerSystem instance for existing instances. Although the key properties are not shown, the
 697 fact that new instances have been created is indicated by the object names used in the diagram.



698

699

Figure 5 – Service Processors after Failover

700 9.2 Reset a Service Processor

701 A client can reset the service processor as follows:

- 702 1) For the given instance of CIM_ComputerSystem, find the associated instance of
- 703 CIM_EnabledLogicalElementCapabilities.
- 704 2) If the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-
- 705 empty array that contains the value 11 (Reset), execute the RequestStateChange() method
- 706 with the value of the RequestedState parameter set to 11 (Reset).

707 The service processor represented by this instance will be disabled and then enabled.

708 9.3 Retrieve the Service Processor Redundancy Status

709 A client can determine the redundancy status for a given instance of CIM_ComputerSystem as follows:

- 710 1) Find the instance of CIM_RedundancySet that is associated with the instance of
711 CIM_ComputerSystem through an instance of CIM_MemberOfCollection.
- 712 2) Retrieve the value of the CIM_RedundancySet.RedundancyStatus property.

713 9.4 Determine Whether Manual Failover Is Supported

714 A client can determine whether a manual failover of the service processor is supported as follows:

- 715 1) Starting with an instance of CIM_ComputerSystem, find an instance of CIM_RedundancySet
716 that is associated with the CIM_ComputerSystem instance through the
717 CIM_MemberOfCollection association.
- 718 2) Find all instances of CIM_IsSpare that reference the CIM_RedundancySet instance. Query the
719 FailoverSupported property of each instance. If the FailoverSupported property of any instance
720 has the value of 3 (Manual) or 4 (Both Manual and Automatic), manual failover is supported.

721 9.5 Force a Service Processor Failover

722 A client can force a failover of the service processor as follows:

- 723 1) Starting with the CIM_ComputerSystem instance to failover from, find the instance of
724 CIM_RedundancySet that is associated with the CIM_ComputerSystem instance through the
725 CIM_MemberOfCollection association.
- 726 2) Find an instance of CIM_ComputerSystem associated with the CIM_RedundancySet instance
727 through the CIM_IsSpare association where the CIM_IsSpare.FailoverSupported property has
728 the value of 3 (Manual) or 4 (Both Manual and Automatic). This instance will be the service
729 processor to failover to.
- 730 3) Invoke the CIM_RedundancySet.Failover() method, specifying the CIM_ComputerSystem
731 instance from step 1) as the value for the FailoverFrom parameter and the
732 CIM_ComputerSystem instance from step 2) as the value for the FailoverTo parameter.

733 9.6 Determine Whether the ElementName Is Modifiable

734 A client can determine whether it can modify the CIM_ComputerSystem.ElementName property as
735 follows:

- 736 1) Find the CIM_EnabledLogicalElementCapabilities instance that is associated with the
737 CIM_ComputerSystem instance.
- 738 2) Query the value of the ElementNameEditSupported property of the
739 CIM_EnabledLogicalElementCapabilities instance. If the value is TRUE, the client can modify
740 the CIM_ComputerSystem.ElementName property.

741 9.7 Determining If State Management Is Supported

742 For a given instance of CIM_ComputerSystem, a client can determine whether state management of the
743 service processor is supported as follows:

- 744 1) Find the CIM_EnabledLogicalElementCapabilities instance that is associated with the
745 CIM_ComputerSystem instance.
- 746 2) Query the value of the RequestedStatesSupported property of the
747 CIM_EnabledLogicalElementCapabilities instance. If at least one value is specified, state
748 management is supported.

749 **10 CIM Elements**

750 Table 15 shows the instances of CIM Elements for this profile. Instances of the CIM Elements shall be
 751 implemented as described in Table 15. Sections 7 (“Implementation”) and 8 (“Methods”) may impose
 752 additional requirements on these elements.

753 **Table 15 – CIM Elements: Service Processor Profile**

Element Name	Requirement	Description
Classes		
CIM_ComputerSystem	Mandatory	See 10.1.
CIM_ElementCapabilities	Conditional	See 10.2.
CIM_EnabledLogicalElementCapabilities	Optional	See 10.3.
CIM_HostedService	Conditional	See 10.4.
CIM_IsSpare	Optional	See 10.5.
CIM_MemberOfCollection	Conditional	See 10.6.
CIM_OwningCollectionElement	Conditional	See 10.7.
CIM_RedundancySet	Optional	See 10.8.
CIM_RegisteredProfile	Mandatory	See 10.9.
CIM_ServiceAffectsElement	Optional	See 10.10.
CIM_TimeService	Optional	See 10.11.
CIM_ManagementController	Optional	See 10.12.
Indications		
None defined in this profile		

754 **10.1 CIM_ComputerSystem**

755 An instance of CIM_ComputerSystem represents each service processor installed in the enclosure.
 756 Table 16 contains the requirements for properties of the instance.

757 **Table 16 – Class: CIM_ComputerSystem**

Elements	Requirement	Notes
Dedicated	Mandatory	Matches 28 (Management Controller) when the service processor is dedicated to a single base system or 29 (Chassis Manager) when the service processor is dedicated to a Modular System.
Name	Mandatory	None
CreationClassName	Mandatory	None
OtherIdentifyingInfo	Optional	This property should be implemented.
IdentifyingDescriptions	Optional	This property should be implemented.
EnabledState	Mandatory	See 7.1.1.
RequestedState	Mandatory	See 7.1.2.2 and 7.1.3.2.
OperationalStatus	Mandatory	None
HealthState	Mandatory	None
ElementName	Mandatory	See 7.1.4 and 7.1.5.
RequestStateChange()	Conditional	See 7.1.2 and 8.1.

758 10.2 CIM_ElementCapabilities

759 CIM_ElementCapabilities associates an instance of CIM_EnabledLogicalElementCapabilities with an
760 instance of CIM_ComputerSystem. Table 17 contains the requirements for properties of the instance.

761 **Table 17 – Class: CIM_ElementCapabilities**

Elements	Requirement	Notes
ManagedElement	Mandatory	This property shall be a reference to an instance of CIM_ComputerSystem. Cardinality 1..*
Capabilities	Mandatory	This property shall be a reference to the instance of CIM_EnabledLogicalElementCapabilities. Cardinality 0..1

762 10.3 CIM_EnabledLogicalElementCapabilities

763 CIM_EnabledLogicalElementCapabilities indicates support for managing the state of the service
764 processor. Table 18 contains the requirements for properties of the instance.

765 **Table 18 – Class: CIM_EnabledLogicalElementCapabilities**

Elements	Requirement	Notes
InstanceID	Mandatory	None
RequestedStatesSupported	Mandatory	See 7.1.2.1.1 and 7.1.3.1.1.
ElementNameEditSupported	Mandatory	See 7.1.4.1.1 and 7.1.5.1.1.
MaxElementNameLen	Conditional	See 7.1.4.1.2 and 7.1.5.1.2.

766 10.4 CIM_HostedService

767 CIM_HostedService relates the CIM_TimeService instance to its scoping CIM_ComputerSystem
768 instance. Table 19 contains the requirements for properties of the instance.

769 **Table 19 – Class: CIM_HostedService**

Elements	Requirement	Notes
Antecedent	Mandatory	This property shall reference the Central Instance. Cardinality 1
Dependent	Mandatory	This property shall reference CIM_TimeService. Cardinality 0..1

770 **10.5 CIM_IsSpare**

771 CIM_IsSpare associates an instance of CIM_ComputerSystem with the CIM_RedundancySet for which
 772 the CIM_ComputerSystem instance represents a spare service processor. Table 20 contains the
 773 requirements for properties of the instance.

774 **Table 20 – Class: CIM_IsSpare**

Elements	Requirement	Description
Antecedent	Mandatory	Reference to the CIM_RedundancySet instance of which the current CIM_ComputerSystem instance is a member and where the CIM_ComputerSystem instance is a spare
Dependent	Mandatory	Reference to the current CIM_ComputerSystem instance
SpareStatus	Optional	See 7.2.4.

775 **10.6 CIM_MemberOfCollection**

776 CIM_MemberOfCollection associates an instance of CIM_ComputerSystem that represents a service
 777 processor with the CIM_RedundancySet of which the CIM_ComputerSystem is a member. Table 21
 778 contains the requirements for properties of the instance.

779 **Table 21 – Class: CIM_MemberOfCollection**

Elements	Requirement	Description
Collection	Mandatory	See 7.2.1. Cardinality 0..1
Member	Mandatory	See 7.2.1. Cardinality *

780 **10.7 CIM_OwningCollectionElement**

781 CIM_OwningCollectionElement associates the CIM_RedundancySet instance with the
 782 CIM_ComputerSystem instance of which the CIM_RedundancySet instance is a member. The instance of
 783 CIM_OwningCollectionElement is conditional on having instantiation of the CIM_RedundancySet class.
 784 Table 22 contains the requirements for properties of the instance.

785 **Table 22 – Class: CIM_OwningCollectionElement**

Elements	Requirement	Notes
OwningElement	Mandatory	See 7.2.2. Cardinality 0..1
OwnedElement	Mandatory	See 7.2.2. Cardinality *

786 **10.8 CIM_RedundancySet**

787 CIM_RedundancySet represents a collection of CIM_ComputerSystem instances that operate as
 788 redundant service processors. Table 23 contains the requirements for properties of the instance.

789 **Table 23 – Class: CIM_RedundancySet**

Elements	Requirement	Notes
InstanceID	Mandatory	None
RedundancyStatus	Mandatory	None
TypeOfSet	Mandatory	See 7.2.
MinNumberNeeded	Mandatory	This property shall match 0 when the minimum number of service processors needed for the redundancy is unknown.
ElementName	Mandatory	This property shall be formatted as a free-form string of variable length (pattern ".*").
Failover()	Optional	See 8.2.

790 **10.9 CIM_RegisteredProfile**

791 CIM_RegisteredProfile identifies the *Service Processor Profile* in order for a client to determine whether
 792 an instance of CIM_ComputerSystem is conformant with this profile. The CIM_RegisteredProfile class is
 793 defined by the *Profile Registration Profile*. With the exception of the mandatory values specified for the
 794 properties in Table 24, the behavior of the CIM_RegisteredProfile instance is in accordance with the
 795 *Profile Registration Profile*.

796 **Table 24 – Class: CIM_RegisteredProfile**

Elements	Requirement	Notes
RegisteredName	Mandatory	This property shall have a value of "Service Processor".
RegisteredVersion	Mandatory	This property shall have a value of "1.1.0".

797 **10.10 CIM_ServiceAffectsElement**

798 CIM_ServiceAffectsElement associates the CIM_TimeService instance with the Central Instance.
 799 Table 25 contains the requirements for properties of the instance.

800 **Table 25 – Class: CIM_ServiceAffectsElement**

Elements	Requirement	Notes
AffectedElement	Mandatory	This property shall be a reference to the Central Instance. Cardinality 1
AffectingElement	Mandatory	This property shall be a reference to an instance of CIM_TimeService. Cardinality 0..1
ElementEffects	Mandatory	Matches 5 (Manages)

801 **10.11 CIM_TimeService**

802 CIM_TimeService manages the current time on the service processor. Table 26 contains the
 803 requirements for properties of the instance.

804 **Table 26 – Class: CIM_TimeService**

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	Key
SystemName	Mandatory	Key
CreationClassName	Mandatory	Key
Name	Mandatory	Key
ElementName	Mandatory	Pattern (".*")

805 **10.12 CIM_ManagementController**

806 CIM_ManagementController is a model construct used by an implementation to support linking the service
 807 processor to other constructions for managing the settings of the service processor, such as PCI or
 808 register information. Table 27 contains the requirements for properties of the instance.

809 **Table 27 – Class: CIM_ManagementController**

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	Key
SystemName	Mandatory	Key
CreationClassName	Mandatory	Key
DeviceID	Mandatory	Key

810

811

812

813
814
815
816

ANNEX A (informative)

Change Log

Version	Date	Description
1.0.0	2009-06-22	DMTF Standard Release
1.1.0	2011-06-30	Added support for the PCI Profile

817