



1

2

3

4

**Document Number: DSP1080**

**Date: 2009-06-16**

**Version: 1.0.0**

5 **Enabled Logical Element Profile**

6 **Document Type: Specification**

7 **Document Status: DMTF Standard**

8 **Document Language: E**

9

## 10 Copyright Notice

11 Copyright © 2007, 2009 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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26 withdrawn or modified after publication, and shall be indemnified and held harmless by any party  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

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

32 **CONTENTS**

33 Foreword ..... 5

34 Introduction ..... 6

35 1 Scope ..... 7

36 2 Normative References..... 7

37 2.1 Approved References ..... 7

38 2.2 Other References..... 7

39 3 Terms and Definitions..... 7

40 4 Symbols and Abbreviated Terms ..... 8

41 5 Synopsis ..... 9

42 6 Description ..... 9

43 6.1 Health State and Operational Status ..... 10

44 7 Implementation Requirements ..... 10

45 7.1 Managing Enabled Logical Element State Is Unsupported ..... 10

46 7.2 Managing Enabled Logical Element State Is Supported ..... 10

47 7.3 CIM\_EnabledLogicalElement.ElementName ..... 12

48 7.4 Representing the Primary Status of the Enabled Logical Element..... 13

49 8 Methods..... 13

50 8.1 Method: CIM\_EnabledLogicalElement.RequestStateChange( ) ..... 13

51 8.2 Profile Conventions for Operations..... 14

52 8.3 CIM\_ElementCapabilities Operations..... 15

53 8.4 CIM\_EnabledLogicalElementCapabilities Operations ..... 15

54 8.5 CIM\_EnabledLogicalElement Operations..... 15

55 9 Use Cases..... 16

56 9.1 General Object Diagram ..... 16

57 9.2 State Transition Object Diagrams..... 17

58 9.3 Determine the Level of State Management Supported ..... 21

59 9.4 Enable the Enabled Logical Element..... 22

60 9.5 Disable the Enabled Logical Element..... 22

61 9.6 Reset the Enabled Logical Element..... 22

62 9.7 Determine Whether the CIM\_EnabledLogicalElement.ElementName Is Modifiable..... 23

63 10 CIM Elements..... 23

64 10.1 CIM\_ElementCapabilities ..... 23

65 10.2 CIM\_EnabledLogicalElementCapabilities..... 24

66 10.3 CIM\_EnabledLogicalElement ..... 24

67 ANNEX A (informative) Change Log..... 25

68

69 **Figures**

70 Figure 1 – Enabled Logical Element Profile: Class Diagram ..... 9

71 Figure 2 – Enabled Logical Element Profile: Object Diagram..... 17

72 Figure 3 – Enabled Logical Element Profile: Enabled State ..... 18

73 Figure 4 – Enabled Logical Element Profile: Transitioning Requested ..... 18

74 Figure 5 – Enabled Logical Element Profile: Transitioning to Disabled State ..... 19

75 Figure 6 – Enabled Logical Element Profile: Disabled State ..... 20

76 Figure 7 – Enabled Logical Element Profile: Transitioning to Enabled State ..... 20

77 Figure 8 – Enabled Logical Element Profile: Transitioned to Enabled State ..... 21

78

79 **Tables**

80 Table 1 – CIM\_EnabledLogicalElement.RequestStateChange( ) Method: Return Code Values..... 14

81 Table 2 – CIM\_EnabledLogicalElement.RequestStateChange( ) Method: Parameters..... 14

82 Table 3 – CIM\_ElementCapabilities Operations ..... 15

83 Table 4 – CIM\_EnabledLogicalElementCapabilities Operations ..... 15

84 Table 5 – CIM\_EnabledLogicalElement Operations..... 15

85 Table 5 – CIM Elements: Enabled Logical Element Profile ..... 23

86 Table 6 – CIM\_ElementCapabilities..... 23

87 Table 7 – CIM\_EnabledLogicalElementCapabilities..... 24

88 Table 8 – Class: CIM\_EnabledLogicalElement..... 24

89

90

## Foreword

91 The *Enabled Logical Element Profile* (DSP1080) was prepared by the CIM Core Working Group.

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

## 94 **Acknowledgments**

95 The authors wish to acknowledge the following people.

### 96 **Editor:**

- 97 • Khachatur Papanyan – Dell Inc.
- 98 • Jon Hass – Dell Inc.

### 99 **Contributors:**

- 100 • Jon Hass – Dell Inc.
- 101 • Joe Kozlowski – Dell Inc.
- 102 • Khachatur Papanyan – Dell Inc.
- 103 • George Ericson – EMC
- 104 • Barb Craig – HP
- 105 • Christina Shaw – HP
- 106 • Jeff Hilland – HP
- 107 • Aaron Merkin – IBM
- 108 • David Hines – Intel
- 109 • John Leung – Intel
- 110 • Steve Hand – Symantec

111

112

## Introduction

113 The information in this specification and referenced specifications should be sufficient for a provider or  
114 consumer of this data to identify unambiguously the classes, properties, methods, and values that shall  
115 be instantiated and manipulated to represent and manage the common aspects of enabled logical  
116 elements that are modeled using the DMTF CIM core and extended model definitions.

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

119

# Enabled Logical Element Profile

## 120 1 Scope

121 The *Enabled Logical Element Profile* extends the management capabilities of referencing profiles by  
122 adding the capability to represent any enabled logical element. The profile describes common  
123 requirements for modeling the variety of enabled logical elements within managed systems including  
124 enabled state management, health state, and operational status.

## 125 2 Normative References

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

### 129 2.1 Approved References

130 DMTF DSP0004, *CIM Infrastructure Specification 2.5*,  
131 [http://www.dmtf.org/standards/published\\_documents/DSP0004\\_2.5.pdf](http://www.dmtf.org/standards/published_documents/DSP0004_2.5.pdf)

132 DMTF DSP0200, *CIM Operations over HTTP 1.2*,  
133 [http://www.dmtf.org/standards/published\\_documents/DSP0200\\_1.2.pdf](http://www.dmtf.org/standards/published_documents/DSP0200_1.2.pdf)

134 DMTF DSP1001, *Management Profile Specification Usage Guide 1.0*,  
135 [http://www.dmtf.org/standards/published\\_documents/DSP1001\\_1.0.pdf](http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf)

### 136 2.2 Other References

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

## 139 3 Terms and Definitions

140 For the purposes of this document, the following terms and definitions apply.

### 141 3.1

#### 142 **can**

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

### 144 3.2

#### 145 **cannot**

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

### 147 3.3

#### 148 **conditional**

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

### 151 3.4

#### 152 **mandatory**

153 indicates requirements to be followed strictly in order to conform to the document and from which no  
154 deviation is permitted

- 155 **3.5**  
156 **may**  
157 indicates a course of action permissible within the limits of the document
- 158 **3.6**  
159 **need not**  
160 indicates a course of action permissible within the limits of the document
- 161 **3.7**  
162 **optional**  
163 indicates a course of action permissible within the limits of the document
- 164 **3.8**  
165 **referencing profile**  
166 indicates a profile that owns the definition of this class and can include a reference to this profile in its  
167 “Related Profiles” table
- 168 **3.9**  
169 **shall**  
170 indicates requirements to be followed strictly in order to conform to the document and from which no  
171 deviation is permitted
- 172 **3.10**  
173 **shall not**  
174 indicates requirements to be followed strictly in order to conform to the document and from which no  
175 deviation is permitted
- 176 **3.11**  
177 **should**  
178 indicates that among several possibilities, one is recommended as particularly suitable, without  
179 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 180 **3.12**  
181 **should not**  
182 indicates that a certain possibility or course of action is deprecated but not prohibited
- 183 **3.13**  
184 **ELE instance**  
185 indicates an instance of CIM concrete class derived from CIM\_EnabledLogicalElement
- 186 **3.14**  
187 **enabled logical element**  
188 logical managed element that has a concept of enabled state associated with it

## 189 **4 Symbols and Abbreviated Terms**

- 190 **4.1**  
191 **CIM**  
192 Common Information Model
- 193 **4.2**  
194 **ELE**  
195 CIM\_EnabledLogicalElement



196 **5 Synopsis**

197 **Profile Name:** Enabled Logical Element

198 **Version:** 1.0.0

199 **Organization:** DMTF

200 **CIM Schema Version:** 2.22

201 **Central Class:** CIM\_EnabledLogicalElement

202 **Scoping Class:** Defined in the specialized profile

203 The *Enabled Logical Element Profile* is an abstract profile that extends the management capability of the  
 204 referencing profiles by adding common representation of enabled logical elements. This abstract profile  
 205 specification shall not be directly implemented; implementations shall be based on a profile specification  
 206 that specializes the requirements of this profile.

207 The *Enabled Logical Element Profile* may be specialized by autonomous profiles and component profiles.

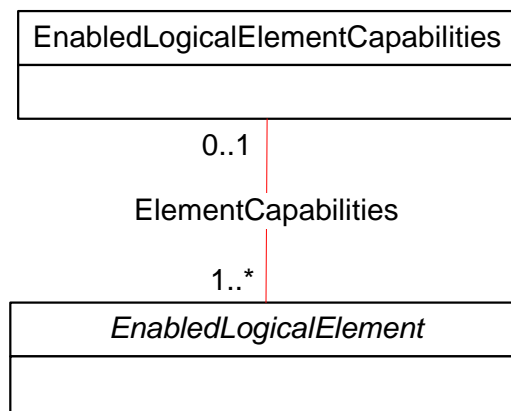
208 The Central Class of the *Enabled Logical Element Profile* shall be ELE (see section 4.2 for the definition  
 209 of ELE). The Central Instance shall be an ELE instance (see section 3.13 for the definition of “ELE  
 210 instance”). The Scoping Class and the Scoping Instance of the *Enabled Logical Element Profile* shall be  
 211 defined in the profile, which specializes from *Enabled Logical Element Profile*.

212 Related profiles are not defined by this standard.

213 **6 Description**

214 The *Enabled Logical Element Profile* is an abstract profile which describes the common set of attributes  
 215 and behavior for enabled logical elements. The profile also specifies a set of properties representing the  
 216 enabled state, the requested state and the current operational and health status of managed elements,  
 217 an optional method for the initiation of enabled state changes, and an optional capability class conveying  
 218 information about supported requested states and the mutability of properties such as the ElementName  
 219 property.

220 Figure 1 represents the class schema for the *Enabled Logical Element Profile*. For simplicity, the prefix  
 221 CIM\_ has been removed from the names of the classes.



222

223

**Figure 1 – Enabled Logical Element Profile: Class Diagram**

224 ELE contains properties to represent the enabled state, different aspects of the operational status, and  
225 health state. CIM\_EnabledLogicalElementCapabilities associated to the ELE through  
226 CIM\_ElementCapabilities represents the capabilities of the associated enabled logical element.

## 227 **6.1 Health State and Operational Status**

228 The health state and operational status for enabled logical element is represented using the following  
229 properties on an ELE instance:

- 230 • HealthState, representing the health state of the enabled logical element
- 231 • PrimaryStatus, representing the primary condition of the enabled logical element such as  
232 commonly used “green”, “yellow”, “red” conditions
- 233 • DetailedStatus, representing more detailed status that is used to expand upon the  
234 PrimaryStatus
- 235 • OperatingStatus, representing succinct information regarding the precise operating status of the  
236 enabled logical element
- 237 • CommunicationStatus, representing status specific to the communications aspects of the  
238 enabled logical element

## 239 **7 Implementation Requirements**

240 Requirements and guidelines for propagating and formulating certain properties of the classes are  
241 discussed in this section. Methods are listed in section 8 and properties are listed in section 10.

### 242 **7.1 Managing Enabled Logical Element State Is Unsupported**

243 If management or representation of the state of the enabled logical element is not supported, the  
244 requirements specified in this clause shall be met.

245 The CIM\_EnabledLogicalElement.RequestedState property shall have the value 12 (Not Applicable). The  
246 CIM\_EnabledLogicalElement.EnabledState property shall have the value 5 (Not Applicable). The  
247 CIM\_EnabledLogicalElement.AvailableRequestedStates property shall be NULL. The  
248 CIM\_EnabledLogicalElement.TransitioningToState property shall be NULL or have the value 12 (Not  
249 Applicable).

250 An instance of CIM\_EnabledLogicalElementCapabilities or its subclass may be instantiated and  
251 associated with the ELE instance. If there is an instance of CIM\_EnabledLogicalElementCapabilities  
252 associated with the ELE instance, the  
253 CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall be NULL.

254 The CIM\_EnabledLogicalElement.RequestStateChange() method shall not be implemented or if  
255 implemented, shall return 1 (Not Supported).

### 256 **7.2 Managing Enabled Logical Element State Is Supported**

257 This clause details the requirements related to representing and managing the state of the enabled logical  
258 element. If management or representation of the state of the enabled logical element is supported, the  
259 requirements specified in this clause shall be met.

#### 260 **7.2.1 General Requirements**

261 The “General Requirements” section details the requirements that are applicable for the enabled logical  
262 state management regardless whether the implementation supports state control of the enabled logical  
263 element through support of the RequestStateChange( ) method.

264 The CIM\_EnabledLogicalElement.RequestedState property shall not have the value 12 (Not Applicable).  
265 The CIM\_EnabledLogicalElement.EnabledState property shall not have the value 5 (Not Applicable). The  
266 CIM\_EnabledLogicalElement.EnabledState shall contain a value which indicates the state of the enabled  
267 logical element. The CIM\_EnabledLogicalElement.RequestedState shall contain a value which indicates  
268 the last requested state of the enabled logical element.

#### 269 **7.2.1.1 Enabled State**

270 The specializing profile may constrain the superset of the CIM\_EnabledLogicalElement.EnabledState  
271 property values and may define the particular interpretation of those values.

272 When the enabled logical element is in transition from one state to another, the EnabledState property is  
273 indeterminate. Thus, if the CIM\_EnabledLogicalElement.TransitioningToState is non-NULL, does not  
274 have the value 5 (No Change) or 12 (Not Applicable) which represents a state transition in progress, the  
275 EnabledState property shall have the value 0 (Unknown).

#### 276 **7.2.1.2 Requested State Transitions**

277 The RequestedState property represents the last requested state of the enabled logical element. If the  
278 implementation cannot represent the last requested state, the RequestedState shall have the value 0  
279 (Unknown).

280 The specializing profile may constrain the superset of the CIM\_EnabledLogicalElement.RequestedState  
281 property values and may define the particular interpretation of those values.

282 The specializing profile may constrain the superset of the RequestedState parameter values for the  
283 CIM\_EnabledLogicalElement.RequestStateChange() method and may define the particular interpretation  
284 of those values.

#### 285 **7.2.1.3 Representing In-Progress Transitions**

286 The CIM\_EnabledLogicalElement.TransitioningToState property may be NULL. If the  
287 CIM\_EnabledLogicalElement.TransitioningToState property is non-NULL, it shall not have the value 12  
288 (Not Applicable).

289 The specializing profile may constrain the superset of the  
290 CIM\_EnabledLogicalElement.TransitioningToState property values and may define the particular  
291 interpretation of those values.

### 292 **7.2.2 Enabled Logical Element State Representation without Control**

293 If representation of the state of the enabled logical element is supported and management of the state  
294 through the CIM\_EnabledLogicalElement.RequestStateChange() method is not supported, the  
295 requirements specified in this clause shall be met.

296 The CIM\_EnabledLogicalElement.AvailableRequestedStates property shall be NULL. If there is an  
297 instance of CIM\_EnabledLogicalElementCapabilities associated with the CIM\_EnabledLogicalElement  
298 instance, the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall be  
299 NULL.

300 The CIM\_EnabledLogicalElement.RequestStateChange() method shall not be implemented or if  
301 implemented, shall return 1 (Unsupported).

### 302 **7.2.3 Enabled Logical Element State Representation with Control**

303 If management of the state of the enabled logical element through the  
304 CIM\_EnabledLogicalElement.RequestStateChange() method is supported, the requirements specified in  
305 this clause shall be met.

### 306 7.2.3.1 Representing Possible Requested States

307 There shall be an instance of CIM\_EnabledLogicalElementCapabilities associated with the  
308 CIM\_EnabledLogicalElement instance. The  
309 CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall contain at least one  
310 value. Each value shall be contained in the  
311 CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property if and only if there exist  
312 conditions under which an invocation of the CIM\_EnabledLogicalElement.RequestStateChange() method  
313 where the RequestedState parameter equals the value returns 0 (Completed with No Error).

314 The specializing profile may constrain the superset of the  
315 CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property values.

### 316 7.2.3.2 Representing Available Requested States

317 The CIM\_EnabledLogicalElement.AvailableRequestedStates property may be NULL which indicates that  
318 the property is not supported.

319 If CIM\_EnabledLogicalElement.AvailableRequestedStates is non-NULL, it shall contain zero or more of  
320 the values contained in the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property  
321 of the instance of CIM\_EnabledLogicalElementCapabilities associated with the  
322 CIM\_EnabledLogicalElement instance, where zero number of values indicates that there are no available  
323 requested states.

324 The CIM\_EnabledLogicalElement.AvailableRequestedStates shall not contain any values that are not  
325 contained in the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property of the  
326 instance of CIM\_EnabledLogicalElementCapabilities associated with the CIM\_EnabledLogicalElement  
327 instance.

328 Each value shall be contained in the CIM\_EnabledLogicalElement.AvailableRequestedStates property  
329 only if an invocation of the CIM\_EnabledLogicalElement.RequestStateChange() method where the  
330 RequestedState parameter equals the value would complete successfully.

## 331 7.3 CIM\_EnabledLogicalElement.ElementName

332 The ElementName property shall be formatted as a free-form string of variable length (pattern “..\*”).

### 333 7.3.1 CIM\_EnabledLogicalElement.ElementName Formulation)

334 The ElementName property should contain the name of the logical device as it would be communicated  
335 to an end-user. The ElementName property should also contain an identifier that can be used by the end-  
336 user to differentiate that logical device from another logical device of the same type contained or  
337 aggregated by the same system. For example, if the logical device is a port on a computer system with  
338 100 ports over sub systems (system 1 and system 2), then the ElementName property could have value  
339 of "port 43 on system 2". If the logical device were a processor on a blade system within a modular  
340 system with two processors per blade system, then the ElementName property could have value of  
341 "processor 2 on system 1".

### 342 7.3.2 Managing CIM\_EnabledLogicalElement.ElementName

343 Client modification of the CIM\_EnabledLogicalElement.ElementName property may be supported. This is  
344 conditional behavior based on the CIM\_EnabledLogicalElementCapabilities.ElementNameEditSupported  
345 property of the instance of CIM\_EnabledLogicalElementCapabilities associated with the  
346 CIM\_EnabledLogicalElement instance.

### 347 **7.3.2.1 Support for the ElementName Property Modification**

348 If client modification of the CIM\_EnabledLogicalElement.ElementName property is supported, the  
349 following requirements shall be met.

350 There shall be an instance of CIM\_EnabledLogicalElementCapabilities associated with the  
351 CIM\_EnabledLogicalElement instance.  
352 CIM\_EnabledLogicalElementCapabilities.ElementNameEditSupported property shall have the value  
353 TRUE. The CIM\_EnabledLogicalElementCapabilities.MaxElementNameLen property shall be non-NULL.  
354 The CIM\_EnabledLogicalElementCapabilities.ElementNameMask property shall contain a regular  
355 expression defined using the syntax specified in Annex C of DSP1001.

### 356 **7.3.2.2 No Support for the ElementName Property Modification**

357 If client modification of the CIM\_EnabledLogicalElement.ElementName is not supported, the  
358 implementation shall comply with either or both of the following requirements:

- 359 • There shall be no instance of CIM\_EnabledLogicalElementCapabilities associated with the  
360 CIM\_EnabledLogicalElement instance.
- 361 • CIM\_EnabledLogicalElementCapabilities.ElementNameEditSupported property shall have the  
362 value FALSE on the instance of CIM\_EnabledLogicalElementCapabilities associated with the  
363 CIM\_EnabledLogicalElement instance.

## 364 **7.4 Representing the Primary Status of the Enabled Logical Element**

365 The CIM\_EnabledLogicalElement.PrimaryStatus property shall be implemented and shall be derived from  
366 the CIM\_EnabledLogicalElement.HealthState using the following algorithm:

- 367 • If the HealthState property value is equal to 0 (Unknown) then the PrimaryStatus property shall  
368 have value 0 (Unknown).
- 369 • If the HealthState property value is equal to 5 (OK) then the PrimaryStatus property shall have  
370 value 1 (OK) corresponding to the commonly used "green" status representation of the  
371 managed element.
- 372 • If the HealthState property value is equal to 10 (Degraded/Warning) or 15 (Minor Failure), then  
373 the PrimaryStatus property shall have value 2 (Degraded) corresponding to the commonly used  
374 "yellow" status representation of the managed element.
- 375 • If the HealthState property value is equal to 20 (Major Failure) or 25 (Critical Failure) or 30  
376 (Non-recoverable Error), then the PrimaryStatus property shall have value 3 (Error)  
377 corresponding to the commonly used "red" status representation of the managed element.

## 378 **8 Methods**

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

### 381 **8.1 Method: CIM\_EnabledLogicalElement.RequestStateChange()**

382 Return values for RequestStateChange() shall be as specified in Table 1 where the method-execution  
383 behavior matches the return-code description. RequestStateChange() method's parameters are specified  
384 in Table 2.

385 Invoking the CIM\_EnabledLogicalElement.RequestStateChange() method multiple times could result in  
386 earlier requests being overwritten or lost.

387 No standard messages are defined for this method.

388 **Table 1 – CIM\_EnabledLogicalElement.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

389 **Table 2 – CIM\_EnabledLogicalElement.RequestStateChange() Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	RequestedState	uint16	Requested state
OUT	Job	CIM_ConcreteJob REF	Returned if job started
IN	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

390 **8.1.1 General Requirements**

391 If the RequestedState parameter is NULL, the CIM\_EnabledLogicalElement.RequestStateChange()  
392 method shall return 2 (Unknown or Unspecified Error).

393 The CIM\_EnabledLogicalElement.RequestStateChange() method shall return 2 (Unknown or Unspecified  
394 Error) if the RequestedState parameter specifies a value that is not listed in the  
395 CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property of the associated instance  
396 of CIM\_EnabledLogicalElementCapabilities.

397 The CIM\_EnabledLogicalElement.RequestStateChange() method shall return 2 (Unknown or Unspecified  
398 Error) if the CIM\_EnabledLogicalElementCapabilities.AvailableRequestedStates property is non-null and  
399 does not contain the value specified by the RequestedState parameter.

400 **8.1.2 Conditional Requirement**

401 If the behavior specified in 7.2.3 is implemented, the CIM\_EnabledLogicalElement.RequestStateChange()  
402 method shall be implemented and shall not return 1 (Not Supported).

403 **8.2 Profile Conventions for Operations**

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

406 The default list of operations is as follows:

- 407 • GetInstance
- 408 • Associators
- 409 • AssociatorNames
- 410 • References
- 411 • ReferenceNames
- 412 • EnumerateInstances
- 413 • EnumerateInstanceNames

414 **8.3 CIM\_ElementCapabilities Operations**

415 Table 3 lists implementation requirements for operations. If implemented, these operations shall be  
 416 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 3, all operations in  
 417 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

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

419 **Table 3 – CIM\_ElementCapabilities Operations**

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

420 **8.4 CIM\_EnabledLogicalElementCapabilities Operations**

421 Table 4 lists implementation requirements for operations. If implemented, these operations shall be  
 422 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 4, all operations in  
 423 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

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

425 **Table 4 – CIM\_EnabledLogicalElementCapabilities Operations**

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

426 **8.5 CIM\_EnabledLogicalElement Operations**

427 Table 5 lists implementation requirements for operations. If implemented, these operations shall be  
 428 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 5, all operations in  
 429 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

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

431 **Table 5 – CIM\_EnabledLogicalElement Operations**

Operation	Requirement	Messages
ModifyInstance	Optional. See 8.5.1.	None
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

432 **8.5.1 CIM\_EnabledLogicalElement—ModifyInstance**

433 This section details the requirements for the ModifyInstance operation applied to an instance of  
 434 CIM\_EnabledLogicalElement. The ModifyInstance operation may be supported.

### 435 **8.5.1.1 General Requirements**

436 The ModifyInstance operation shall be supported and the CIM\_EnabledLogicalElement.ElementName  
437 property shall be modifiable when an instance of CIM\_EnabledLogicalElementCapabilities is associated  
438 with the CIM\_EnabledLogicalElement instance and the ElementNameEditSupported property of the  
439 CIM\_EnabledLogicalElementCapabilities instance associated with the CIM\_EnabledLogicalElement  
440 instance has a value of TRUE. See 8.5.1.2.

### 441 **8.5.1.2 CIM\_EnabledLogicalElement.ElementName**

442 If an instance of CIM\_EnabledLogicalElementCapabilities is associated with the  
443 CIM\_EnabledLogicalElement instance and the ElementNameEditSupported property of the  
444 CIM\_EnabledLogicalElementCapabilities instance associated with the CIM\_EnabledLogicalElement  
445 instance has a value of TRUE, the implementation shall allow the ModifyInstance operation to change the  
446 value of the ElementName property of the CIM\_EnabledLogicalElement instance. The ModifyInstance  
447 operation shall enforce the length restriction specified in the MaxElementNameLen property of the  
448 CIM\_EnabledLogicalElementCapabilities instance. The ModifyInstance operation shall enforce the  
449 regular expression specified in the ElementNameMask property of the  
450 CIM\_EnabledLogicalElementCapabilities instance.

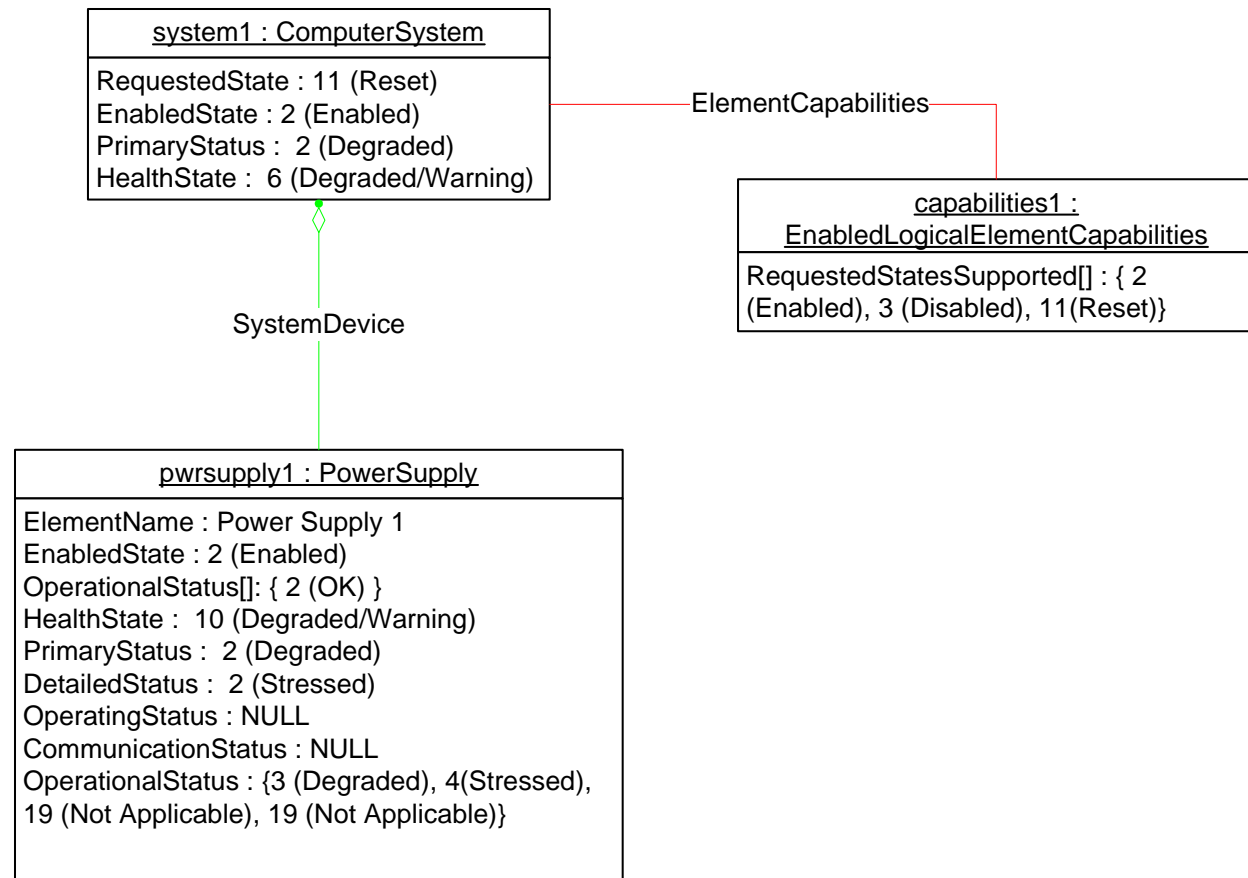
## 451 **9 Use Cases**

452 This section contains object diagrams and use cases for the *Enabled Logical Element Profile*.

### 453 **9.1 General Object Diagram**

454 Figure 2 represents an instantiation of enabled logical elements conforming with the *Enabled Logical*  
455 *Element Profile*. System1 supports the state management feature and per capabilities1 could be enabled,  
456 disabled and reset. System1 has been previously reset per the RequestedState property having value of  
457 11 (Reset) but is currently enabled with degraded status. Pwrsupply1 is also degraded per the  
458 PrimaryStatus property which is correctly derived from the HealthState property. Pwrsupply1 also reports  
459 more granular status with the DetailedStatus property and accumulated statuses are represented in the  
460 OperationalStatus array property.





461

462

**Figure 2 – Enabled Logical Element Profile: Object Diagram**

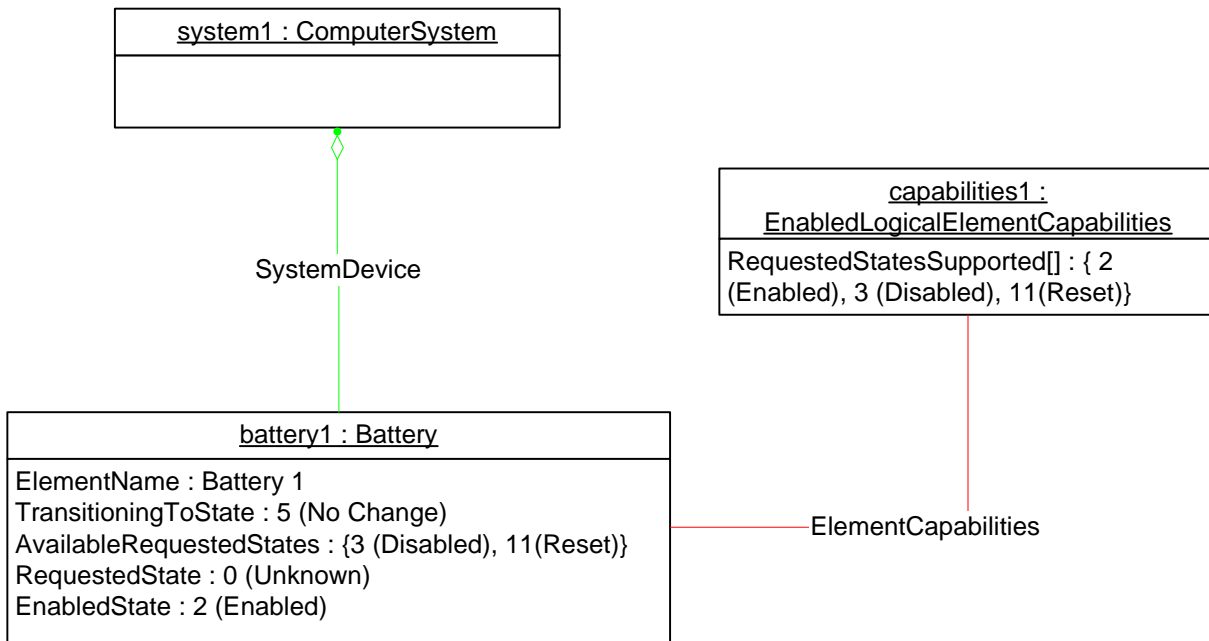
463 **9.2 State Transition Object Diagrams**

464 Figure 3 represents an instantiation of an enabled logical element conforming with the *Enabled Logical*  
 465 *Element Profile*. battery1 represents an enabled logical element with the EnabledState property set  
 466 2(Enabled) representing that the battery is currently enabled. The RequestedState property set to 0  
 467 (Unknown) represents that the last requested state transition for battery1 is unknown.  
 468 AvailableRequestedChange array contains the current state transitions supported for battery1 at its  
 469 current state. Note that capabilities1's the RequestedStatesSupported property advertises all the state  
 470 transitions possible for battery1 regardless of its current state.

471 Battery1 is currently not in transition to any state since the TransitionToState property is set to 5 (No  
 472 Change). But a transition could be initiated by executing the RequestedStateChange() method.

473 Sections 9.2.1, 9.2.2, 9.2.3, 9.2.4 and 9.2.5 describe the different states that battery1 could be after the  
 474 successful execution of the RequestStateChange() method with the RequestedState parameter set to  
 475 11(Reset), regardless whether the state transitions are synchronous or asynchronous of the method  
 476 execution.

477 NOTE: Capabilities1's RequestedStatesSupported property does not change regardless of the current state of  
 478 battery1, in contrast to battery1's AvailableRequestedStates property that changes depending on the state  
 479 of battery1.



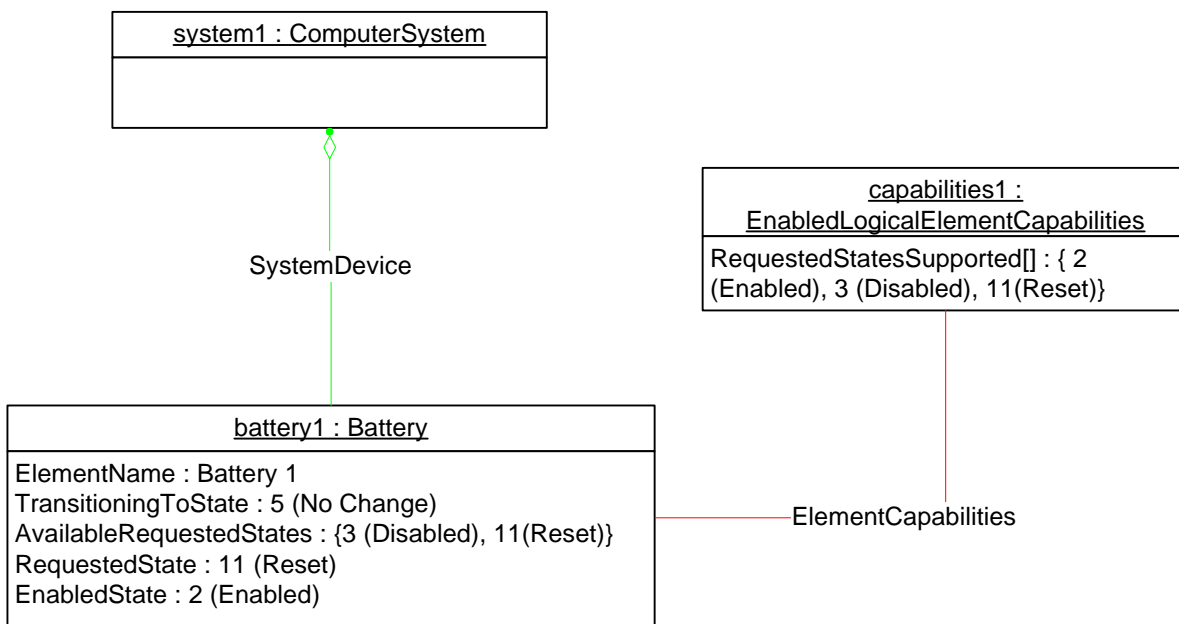
480

481

**Figure 3 – Enabled Logical Element Profile: Enabled State**

**482 9.2.1 Successful Transitioning Request**

483 Figure 4 shows battery1 has successfully received the state transitioning request to 11 (Reset) as a result  
 484 of the successful execution of the RequestStateChange() method with the RequestedState parameter set  
 485 to 11 (Reset) as represented by the last requested state RequestedState property value of 11 (Reset).  
 486 Battery1's EnabledState has a value of 2 (Enabled) and TransitioningToState property has a value of 5  
 487 (No Change), representing that battery1 is currently enabled and has not yet started the state transition.



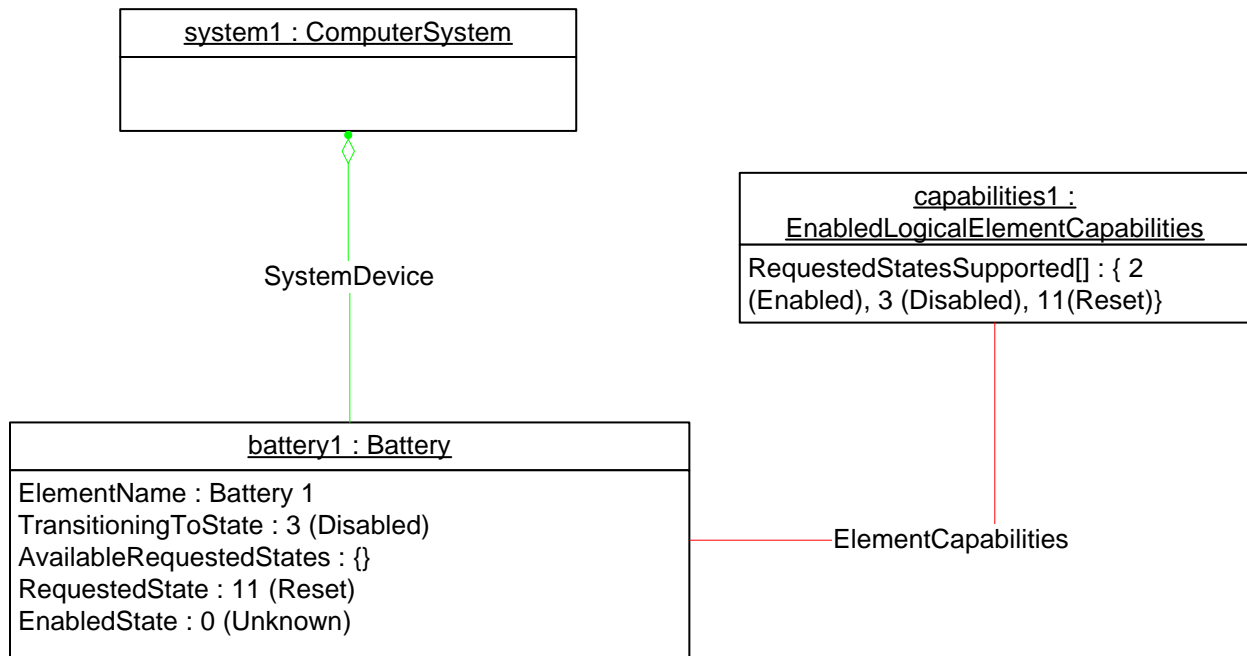
488

489

**Figure 4 – Enabled Logical Element Profile: Transitioning Requested**

490 **9.2.2 Transitioning to Disabled State**

491 Figure 5 shows battery1 in transition state to disabled as a result of the successful execution of the  
 492 RequestStateChange() method with the RequestedState parameter set to 11 (Reset), as represented by  
 493 the last requested state RequestedState property value of 11 (Reset). Battery1's EnabledState has a  
 494 value of 0 (Unknown) and TransitioningToState property has a value of 3 (Disabled), representing that  
 495 battery1 is currently in transition to the disabled state. The AvailableRequestedStates property is an  
 496 empty array representing that the implementation does not accept any state change requests at this  
 497 particular time.

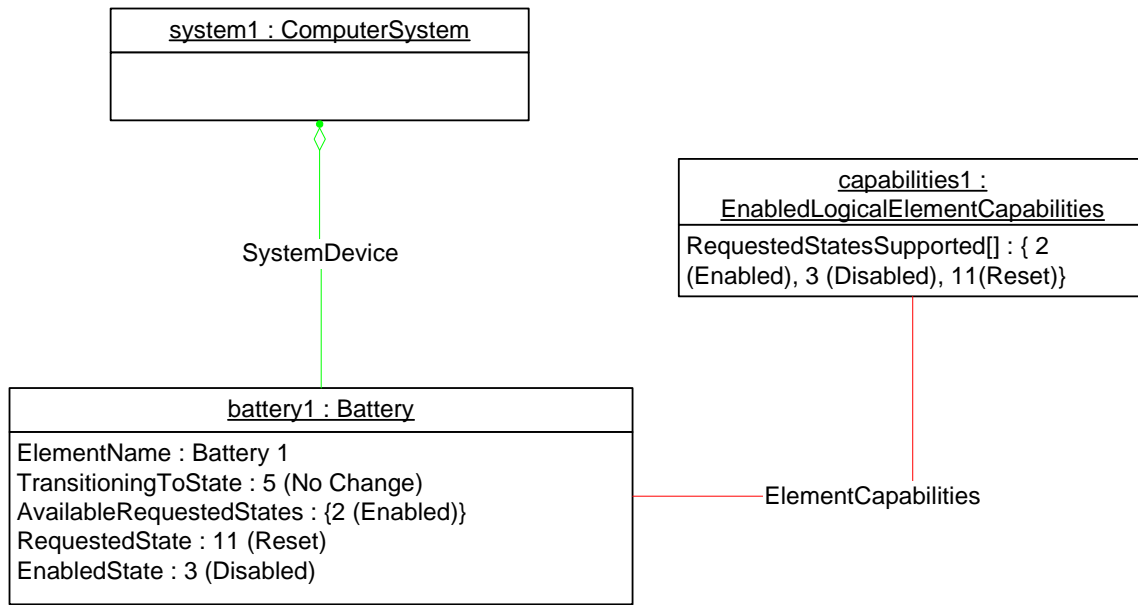


498

499 **Figure 5 – Enabled Logical Element Profile: Transitioning to Disabled State**

500 **9.2.3 Transitioned to Disabled State**

501 Figure 6 shows battery1 as it transitioned to the disabled state as a result of the successful execution of  
 502 the RequestStateChange() method with the RequestedState parameter set to 11(Reset) as represented  
 503 by the last requested state RequestedState property value of 11 (Reset). Battery1's EnabledState has a  
 504 value of 3 (Disabled) and TransitioningToState property has a value of 5 (No Change), representing that  
 505 battery1 is currently in the disabled state. The AvailableRequestedStates property contains the value 2  
 506 (Enabled), representing that the implementation accepts the state change request to enable battery1 at  
 507 this particular time.



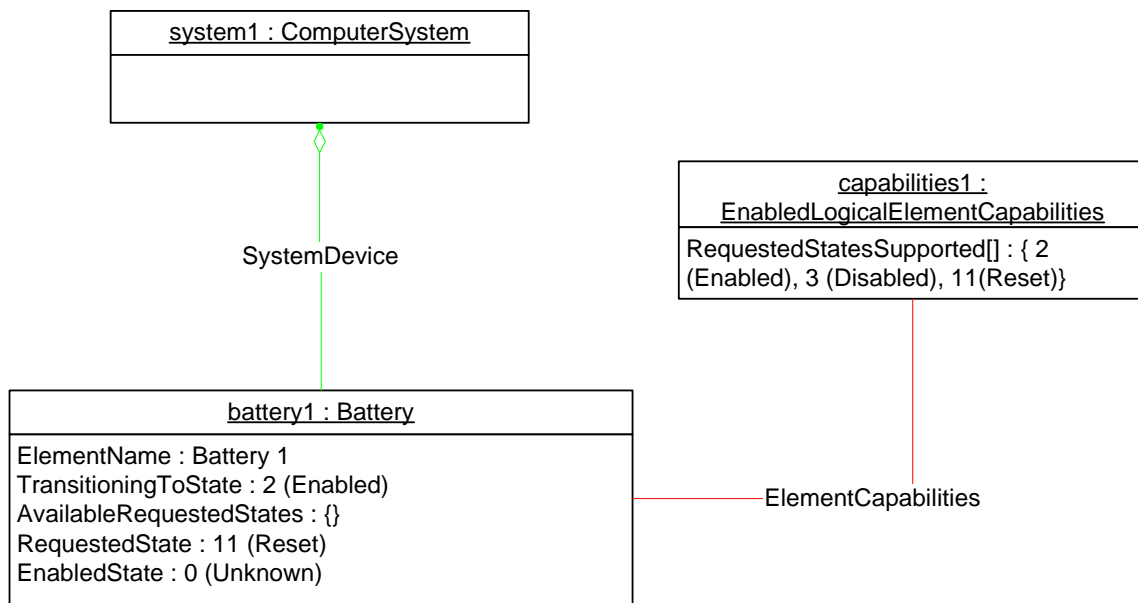
508

509

**Figure 6 – Enabled Logical Element Profile: Disabled State**

510 **9.2.4 Transitioning to Enabled State**

511 Figure 7 shows battery1 in transition state to enabled as a result of the successful execution of the  
 512 RequestStateChange() method with the RequestedState parameter set to 11 (Reset) as represented by  
 513 the last requested state RequestedState property value of 11 (Reset). Battery1’s EnabledState has a  
 514 value of 0 (Unknown) and TransitioningToState property has a value of 2 (Enabled) representing that  
 515 battery1 is currently in transition to the enabled state. The AvailableRequestedStates property is an  
 516 empty array representing that the implementation does not accept any state change requests at this  
 517 particular time.



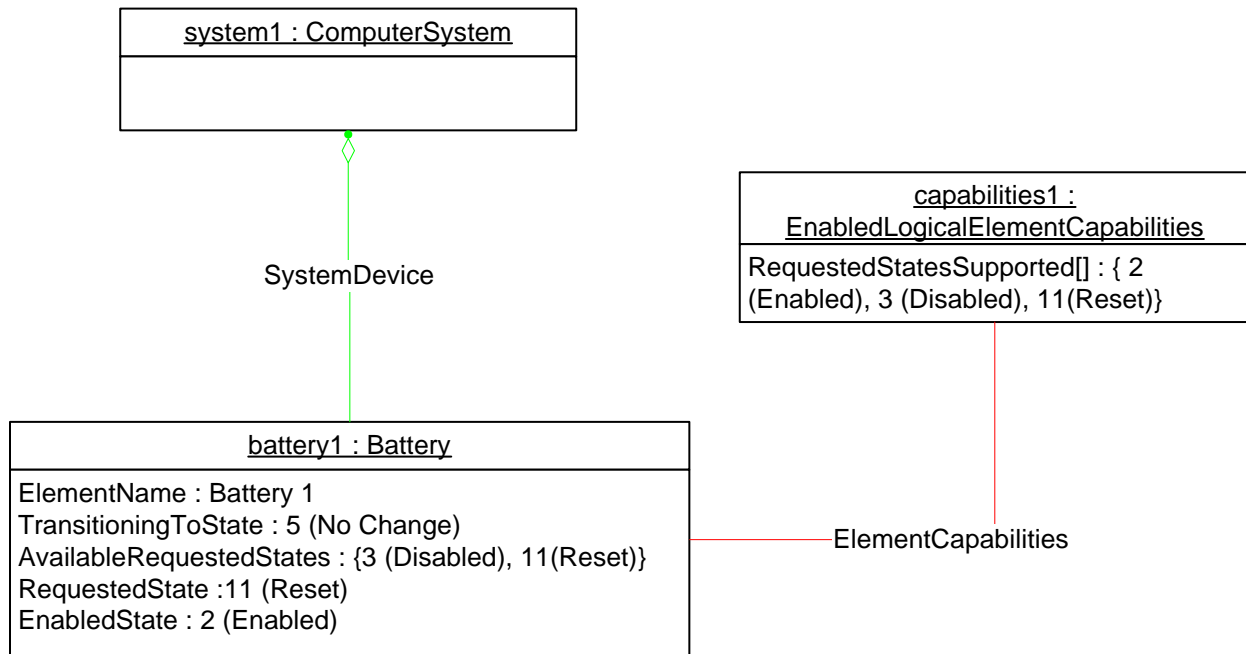
518

519

**Figure 7 – Enabled Logical Element Profile: Transitioning to Enabled State**

520 **9.2.5 Transitioned to Enabled**

521 Figure 8 shows battery1’s final state as a result of the successful execution of the RequestStateChange()  
 522 method with the RequestedState parameter set to 11 (Reset) as represented by the last requested state  
 523 RequestedState property value of 11 (Reset). Battery1’s EnabledState has a value of 2 (Enabled) and  
 524 TransitioningToState property has a value of 5 (No Change), representing that battery1 is currently in the  
 525 enabled state. The AvailableRequestedStates property contains values 3 (Disabled) and 11 (Reset),  
 526 representing that the implementation accepts disabling or resetting battery1 at this particular time.



527

528 **Figure 8 – Enabled Logical Element Profile: Transitioned to Enabled State**

529 **9.3 Determine the Level of State Management Supported**

530 A client can determine the level of the state management supported by the enabled logical element as  
 531 follows:

- 532 1) For the given ELE instance, retrieve the EnabledState and RequestedState properties.
- 533 2) If the EnabledState and RequestedState properties do not have the value of 12 (Not  
 534 Applicable), then the representation of state management is supported; continue to step 3.  
 535 Otherwise, neither the representation of state management nor the state management control  
 536 for the enabled logical element is supported.
- 537 3) Find the associated instance of CIM\_EnabledLogicalElementCapabilities.
- 538 4) If the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-  
 539 empty array, then the state management control for the enabled logical element is supported as  
 540 well.

## 541 9.4 Enable the Enabled Logical Element

542 A client can enable the enabled logical element as follows:

- 543 1) For the given ELE instance, find the associated instance of  
544 CIM\_EnabledLogicalElementCapabilities.
- 545 2) If the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-  
546 empty array and contains the value 2 (Enabled), continue to step 3; otherwise, the  
547 instrumentation does not support enabling the enabled logical element.
- 548 3) If the given ELE instance's AvailableRequestedStates property is a non-NULL or non-empty  
549 array and contains 2 (Enabled), continue to step 4; otherwise, the instrumentation supports  
550 enabling the enabled logical element but cannot transition to 2 (Enabled) state at this time.
- 551 4) Execute the RequestStateChange() method with the value of the RequestedState parameter  
552 set to 2 (Enabled), which requests to enable the enabled logical element.
- 553 5) If the RequestStateChange() method execution returns 0 (Success), the instrumentation has  
554 successfully processed the request to transition the enabled logical element's state to 2  
555 (Enabled).

## 556 9.5 Disable the Enabled Logical Element

557 A client can disable the enabled logical element as follows:

- 558 1) For the given ELE instance, find the associated instance of  
559 CIM\_EnabledLogicalElementCapabilities.
- 560 2) If the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-  
561 empty array and contains the value 3 (Disabled), continue to step 3; otherwise, the  
562 instrumentation does not support enabling the enabled logical element.
- 563 3) If the given ELE instance's AvailableRequestedStates property is a non-NULL or non-empty  
564 array and contains 3 (Disabled), continue to step 4; otherwise, the instrumentation supports  
565 disabling the enabled logical element but cannot transition to 3 (Disabled) state at this time.
- 566 4) Execute the RequestStateChange() method with the value of the RequestedState parameter  
567 set to 3 (Disabled), which requests to disable the enabled logical element.
- 568 5) If the RequestStateChange() method execution returns 0 (Success), the instrumentation has  
569 successfully processed the request to transition the enabled logical element's state to 3  
570 (Disabled).

## 571 9.6 Reset the Enabled Logical Element

572 A client can reset the enabled logical element as follows:

- 573 1) For the given ELE instance, find the associated instance of  
574 CIM\_EnabledLogicalElementCapabilities.
- 575 2) If the CIM\_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-  
576 empty array and contains the value 11 (Reset), continue to step 3; otherwise, the  
577 instrumentation does not support enabling the enabled logical element.
- 578 3) If the given ELE instance's AvailableRequestedStates property is a non-NULL or non-empty  
579 array and contains 11 (Reset), continue to step 4; otherwise, the instrumentation supports  
580 resetting the enabled logical element but cannot perform the reset transition state at this time.
- 581 4) Execute the RequestStateChange() method with the value of the RequestedState parameter  
582 set to 11 (Reset), which requests to reset the enabled logical element.
- 583 5) If the RequestStateChange() method execution returns 0 (Success), the instrumentation has  
584 successfully processed the request to reset the enabled logical element.

585 **9.7 Determine Whether the CIM\_EnabledLogicalElement.ElementName Is**  
 586 **Modifiable**

587 A client can determine whether it can modify the CIM\_EnabledLogicalElement.ElementName property as  
 588 follows:

- 589 1) Find the CIM\_EnabledLogicalElementCapabilities instance that is associated with the ELE  
 590 instance.
- 591 2) Query the value of the ElementNameEditSupported property of the instance. If the value is  
 592 TRUE, the client can modify the CIM\_EnabledLogicalElement.ElementName property.

593 **10 CIM Elements**

594 Table 6 shows the instances of CIM Elements for this profile. Instances of the CIM Elements shall be  
 595 implemented as described in Table 6. Sections 7 (“Implementation Requirements”) and 8 (“Methods”)  
 596 may impose additional requirements on these elements.

597 **Table 6 – CIM Elements: Enabled Logical Element Profile**

Element Name	Requirement	Description
<b>Classes</b>		
CIM_ElementCapabilities	Conditional	See 10.1.
CIM_EnabledLogicalElementCapabilities	Optional	See 7.1, 7.2, and 10.2.
CIM_EnabledLogicalElement	Mandatory	See 10.3.
<b>Indications</b>		
None defined in this profile		

598 **10.1 CIM\_ElementCapabilities**

599 CIM\_ElementCapabilities is used to associate an ELE instance with an instance of  
 600 CIM\_EnabledLogicalElementCapabilities that describes the capabilities of the ELE instance.  
 601 CIM\_ElementCapabilities is mandatory if the CIM\_EnabledLogicalElementCapabilities instance is  
 602 instantiated.

603 **Table 7 – CIM\_ElementCapabilities**

Properties	Requirement	Notes
ManagedElement	Mandatory	<b>Key:</b> Shall reference the ELE instance Cardinality 1..* indicating one or more references
Capabilities	Mandatory	<b>Key:</b> Shall reference the instance of CIM_EnabledLogicalElementCapabilities Cardinality 0..1 indicating zero or one reference

604 **10.2 CIM\_EnabledLogicalElementCapabilities**

605 CIM\_EnabledLogicalElementCapabilities represents the capabilities of the enabled logical element.

606 **Table 8 – CIM\_EnabledLogicalElementCapabilities**

Properties	Requirement	Notes
InstanceID	Mandatory	<b>Key</b>
RequestedStatesSupported	Optional	See 7.1 and 7.2.
ElementNameEditSupported	Mandatory	See 7.3.2.
MaxElementNameLen	Conditional	See 7.3.2.
ElementNameMask	Conditional	See 7.3.2.

607 **10.3 CIM\_EnabledLogicalElement**

608 CIM\_EnabledLogicalElement is an abstract class that is used to represent any enabled logical element.

609 **Table 9 – Class: CIM\_EnabledLogicalElement**

Properties and Methods	Requirement	Description
ElementName	Mandatory	See 7.3.
PrimaryStatus	Mandatory	See 7.4.
DetailedStatus	Optional	See 7.4.
OperatingStatus	Optional	See 7.4.
CommunicationStatus	Optional	See 7.4.
HealthState	Mandatory	See 7.4.
EnabledState	Mandatory	See 7.1 and 7.2.
RequestedState	Mandatory	See 7.1 and 7.2.
AvailableRequestedStates	Optional	See 7.1 and 7.2.
TransitioningToState	Optional	See 7.1 and 7.2.
RequestStateChange( )	Conditional	See 8.1.

610



611  
612  
613  
614

**ANNEX A**  
(informative)  
**Change Log**

Version	Date	Description
1.0.0	6/16/2009	DMTF Standard Release

615  
616