Document Number: DSP1080

Date: 2009-06-16

Version: 1.0.0

Enabled Logical Element Profile

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Figures

79 # Tables

89

Foreword

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

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

Acknowledgments

The authors wish to acknowledge the following people.

Editor:

- Khachatur Papanyan – Dell Inc.
- Jon Hass – Dell Inc.

Contributors:

- Jon Hass – Dell Inc.
- Joe Kozlowski – Dell Inc.
- Khachatur Papanyan – Dell Inc.
- George Ericson – EMC
- Barb Craig – HP
- Christina Shaw – HP
- Jeff Hilland – HP
- Aaron Merkin – IBM
- David Hines – Intel
- John Leung – Intel
- Steve Hand – Symantec

Introduction

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

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

Enabled Logical Element Profile

1 Scope

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

2 Normative References

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

2.1 Approved References

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

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

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

2.2 Other References

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

3 Terms and Definitions

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

3.1
can
used for statements of possibility and capability, whether material, physical, or causal

3.2
cannot
used for statements of possibility and capability, whether material, physical, or causal

3.3
conditional
indicates requirements to be followed strictly in order to conform to the document when the specified conditions are met

3.4
mandatory
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.5
may
indicates a course of action permissible within the limits of the document

3.6
need not
indicates a course of action permissible within the limits of the document

3.7
optional
indicates a course of action permissible within the limits of the document

3.8
referencing profile
indicates a profile that owns the definition of this class and can include a reference to this profile in its "Related Profiles" table

3.9
shall
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.10
shall not
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.11
should
indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

3.12
should not
indicates that a certain possibility or course of action is deprecated but not prohibited

3.13
ELE instance
indicates an instance of CIM concrete class derived from CIM_EnabledLogicalElement

3.14
enabled logical element

logical managed element that has a concept of enabled state associated with it

4 Symbols and Abbreviated Terms

4.1
CIM
Common Information Model

4.2
ELE
CIM_EnabledLogicalElement

5 Synopsis

Profile Name: Enabled Logical Element

Version: 1.0.0

Organization: DMTF

CIM Schema Version: 2.22

Central Class: CIM_EnabledLogicalElement

Scoping Class: Defined in the specialized profile

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

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

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

Related profiles are not defined by this standard.

6 Description

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

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

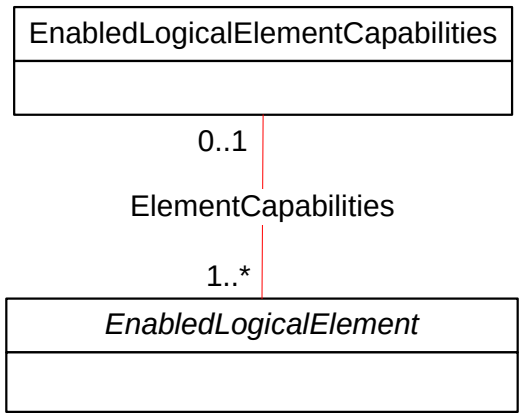

Figure 1 – Enabled Logical Element Profile: Class Diagram

ELE contains properties to represent the enabled state, different aspects of the operational status, and health state. CIM_EnabledLogicalElementCapabilities associated to the ELE through CIM_ElementCapabilities represents the capabilities of the associated enabled logical element.

6.1 Health State and Operational Status

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

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

7 Implementation Requirements

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

7.1 Managing Enabled Logical Element State Is Unsupported

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

The CIM_EnabledLogicalElement.RequestedState property shall have the value 12 (Not Applicable). The CIM_EnabledLogicalElement.EnabledState property shall have the value 5 (Not Applicable). The CIM_EnabledLogicalElement.AvailableRequestedStates property shall be NULL. The CIM_EnabledLogicalElement.TransitioningToState property shall be NULL or have the value 12 (Not Applicable).

An instance of CIM_EnabledLogicalElementCapabilities or its subclass may be instantiated and associated with the ELE instance. If there is an instance of CIM_EnabledLogicalElementCapabilities associated with the ELE instance, the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall be NULL.

The CIM_EnabledLogicalElement.RequestStateChange() method shall not be implemented or if implemented, shall return 1 (Not Supported).

7.2 Managing Enabled Logical Element State Is Supported

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

7.2.1 General Requirements

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

The CIM_EnabledLogicalElement.RequestedState property shall not have the value 12 (Not Applicable). The CIM_EnabledLogicalElement.EnabledState property shall not have the value 5 (Not Applicable). The CIM_EnabledLogicalElement.EnabledState shall contain a value which indicates the state of the enabled logical element. The CIM_EnabledLogicalElement.RequestedState shall contain a value which indicates the last requested state of the enabled logical element.

7.2.1.1 Enabled State

The specializing profile may constrain the superset of the CIM_EnabledLogicalElement.EnabledState property values and may define the particular interpretation of those values.

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

7.2.1.2 Requested State Transitions

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

The specializing profile may constrain the superset of the CIM_EnabledLogicalElement.RequestedState property values and may define the particular interpretation of those values.

The specializing profile may constrain the superset of the RequestedState parameter values for the CIM_EnabledLogicalElement.RequestStateChange() method and may define the particular interpretation of those values.

7.2.1.3 Representing In-Progress Transitions

The CIM_EnabledLogicalElement.TransitioningToState property may be NULL. If the CIM_EnabledLogicalElement.TransitioningToState property is non-NULL, it shall not have the value 12 (Not Applicable).

The specializing profile may constrain the superset of the CIM_EnabledLogicalElement.TransitioningToState property values and may define the particular interpretation of those values.

7.2.2 Enabled Logical Element State Representation without Control

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

The CIM_EnabledLogicalElement.AvailableRequestedStates property shall be NULL. If there is an instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance, the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall be NULL.

The CIM_EnabledLogicalElement.RequestStateChange() method shall not be implemented or if implemented, shall return 1 (Unsupported).

7.2.3 Enabled Logical Element State Representation with Control

If management of the state of the enabled logical element through the CIM_EnabledLogicalElement.RequestStateChange() method is supported, the requirements specified in this clause shall be met.

7.2.3.1 Representing Possible Requested States

There shall be an instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance. The CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall contain at least one value. Each value shall be contained in the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property if and only if there exist conditions under which an invocation of the CIM_EnabledLogicalElement.RequestStateChange() method where the RequestedState parameter equals the value returns 0 (Completed with No Error).

The specializing profile may constrain the superset of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property values.

7.2.3.2 Representing Available Requested States

The CIM_EnabledLogicalElement.AvailableRequestedStates property may be NULL which indicates that the property is not supported.

If CIM_EnabledLogicalElement.AvailableRequestedStates is non-NULL, it shall contain zero or more of the values contained in the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property of the instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance, where zero number of values indicates that there are no available requested states.

The CIM_EnabledLogicalElement.AvailableRequestedStates shall not contain any values that are not contained in the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property of the instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance.

Each value shall be contained in the CIM_EnabledLogicalElement.AvailableRequestedStates property only if an invocation of the CIM_EnabledLogicalElement.RequestStateChange() method where the RequestedState parameter equals the value would complete successfully.

7.3 CIM_EnabledLogicalElement.ElementName

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

7.3.1 CIM_EnabledLogicalElement.ElementName Formulation)

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

7.3.2 Managing CIM_EnabledLogicalElement.ElementName

Client modification of the CIM_EnabledLogicalElement.ElementName property may be supported. This is conditional behavior based on the CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported property of the instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance.

7.3.2.1 Support for the ElementName Property Modification

If client modification of the CIM_EnabledLogicalElement.ElementName property is supported, the following requirements shall be met.

There shall be an instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance.
CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported property shall have the value TRUE. The CIM_EnabledLogicalElementCapabilities.MaxElementNameLen property shall be non-NULL. The CIM_EnabledLogicalElementCapabilities.ElementNameMask property shall contain a regular expression defined using the syntax specified in Annex C of DSP1001.

7.3.2.2 No Support for the ElementName Property Modification

If client modification of the CIM_EnabledLogicalElement.ElementName is not supported, the implementation shall comply with either or both of the following requirements:

- There shall be no instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance.
- CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported property shall have the value FALSE on the instance of CIM_EnabledLogicalElementCapabilities associated with the CIM_EnabledLogicalElement instance.

7.4 Representing the Primary Status of the Enabled Logical Element

The CIM_EnabledLogicalElement.PrimaryStatus property shall be implemented and shall be derived from the CIM_EnabledLogicalElement.HealthState using the following algorithm:

- If the HealthState property value is equal to 0 (Unkown) then the PrimaryStatus property shall have value 0 (Unknown).
- If the HealthState property value is equal to 5 (OK) then the PrimaryStatus property shall have value 1 (OK) corresponding to the commonly used “green” status representation of the managed element.
- If the HealthState property value is equal to 10 (Degraded/Warning) or 15 (Minor Failure), then the PrimaryStatus property shall have value 2 (Degraded) corresponding to the commonly used “yellow” status representation of the managed element.
- If the HealthState property value is equal to 20 (Major Failure) or 25 (Critical Failure) or 30 (Non-recoverable Error), then the PrimaryStatus property shall have value 3 (Error) corresponding to the commonly used “red” status representation of the managed element.

8 Methods

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

8.1 Method: CIM_EnabledLogicalElement.RequestStateChange( )

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

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

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

8.3 CIM_ElementCapabilities Operations

Table 3 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 3, all operations in the default list in 8.2 shall be implemented as defined in DSP0200.

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

Table 3 – CIM_ElementCapabilities Operations

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

8.4 CIM_EnabledLogicalElementCapabilities Operations

Table 4 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 4, all operations in the default list in 8.2 shall be implemented as defined in DSP0200.

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

Table 4 – CIM_EnabledLogicalElementCapabilities Operations

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

8.5 CIM_EnabledLogicalElement Operations

Table 5 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 5, all operations in the default list in 8.2 shall be implemented as defined in DSP0200.

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

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

8.5.1 CIM_EnabledLogicalElement—ModifyInstance

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

8.5.1.1 General Requirements

The ModifyInstance operation shall be supported and the CIM_EnabledLogicalElement.ElementName property shall be modifiable when an instance of CIM_EnabledLogicalElementCapabilities is associated with the CIM_EnabledLogicalElement instance and the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance associated with the CIM_EnabledLogicalElement instance has a value of TRUE. See 8.5.1.2.

8.5.1.2 CIM_EnabledLogicalElement.ElementName

If an instance of CIM_EnabledLogicalElementCapabilities is associated with the CIM_EnabledLogicalElement instance and the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance associated with the CIM_EnabledLogicalElement instance has a value of TRUE, the implementation shall allow the ModifyInstance operation to change the value of the ElementName property of the CIM_EnabledLogicalElement instance. The ModifyInstance operation shall enforce the length restriction specified in the MaxElementNameLen property of the CIM_EnabledLogicalElementCapabilities instance. The ModifyInstance operation shall enforce the regular expression specified in the ElementNameMask property of the CIM_EnabledLogicalElementCapabilities instance.

9 Use Cases

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

9.1 General Object Diagram

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

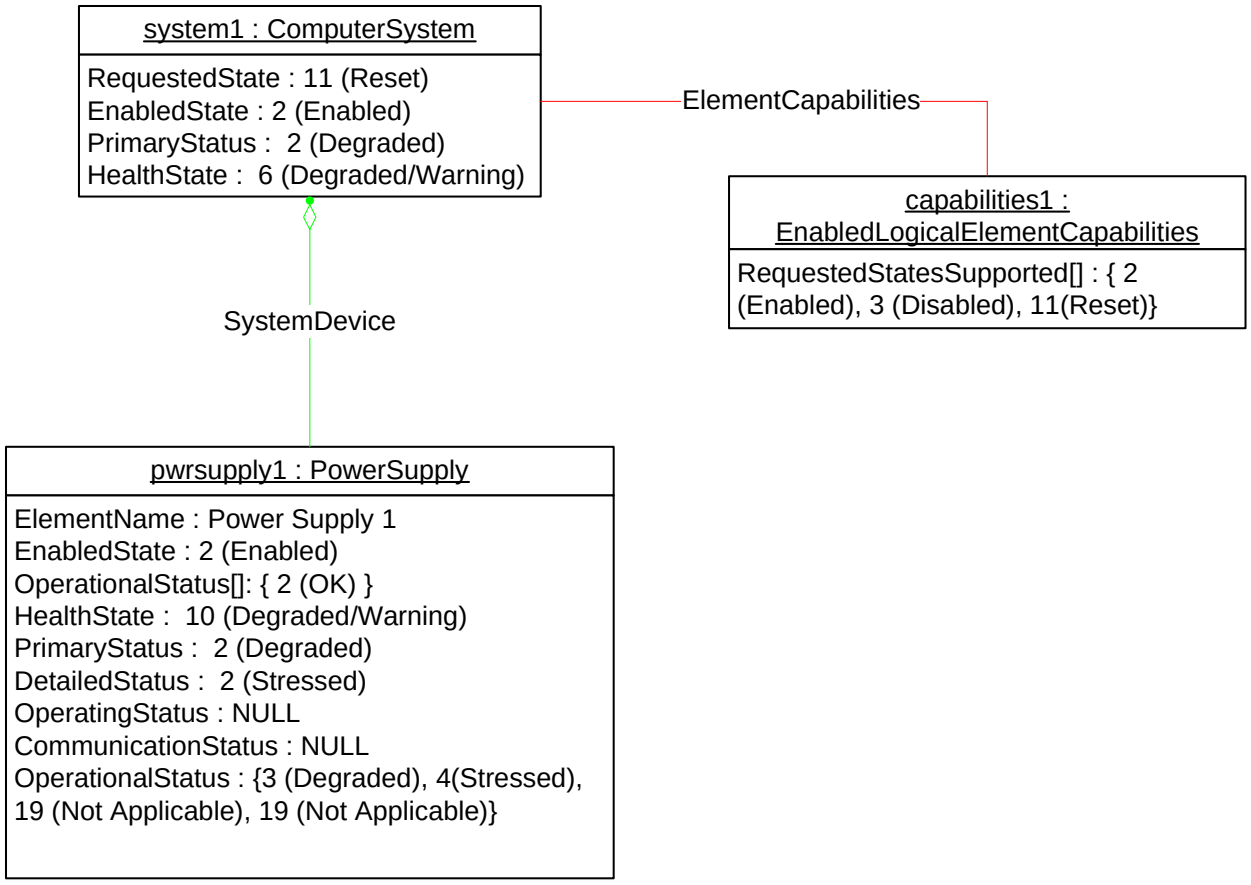

Figure 2 – Enabled Logical Element Profile: Object Diagram

9.2 State Transition Object Diagrams

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

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

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 successful execution of the RequestStateChange() method with the RequestedState parameter set to 11(Reset), regardless whether the state transitions are synchronous or asynchronous of the method execution.

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

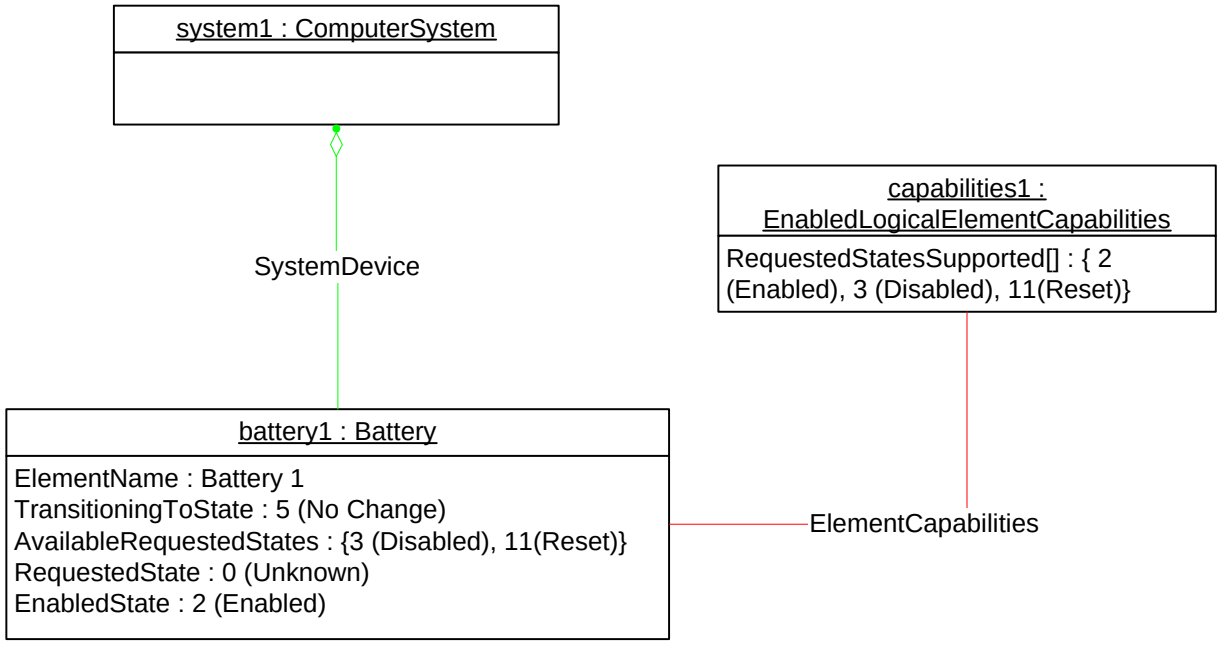

Figure 3 – Enabled Logical Element Profile: Enabled State

9.2.1 Successful Transitioning Request

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

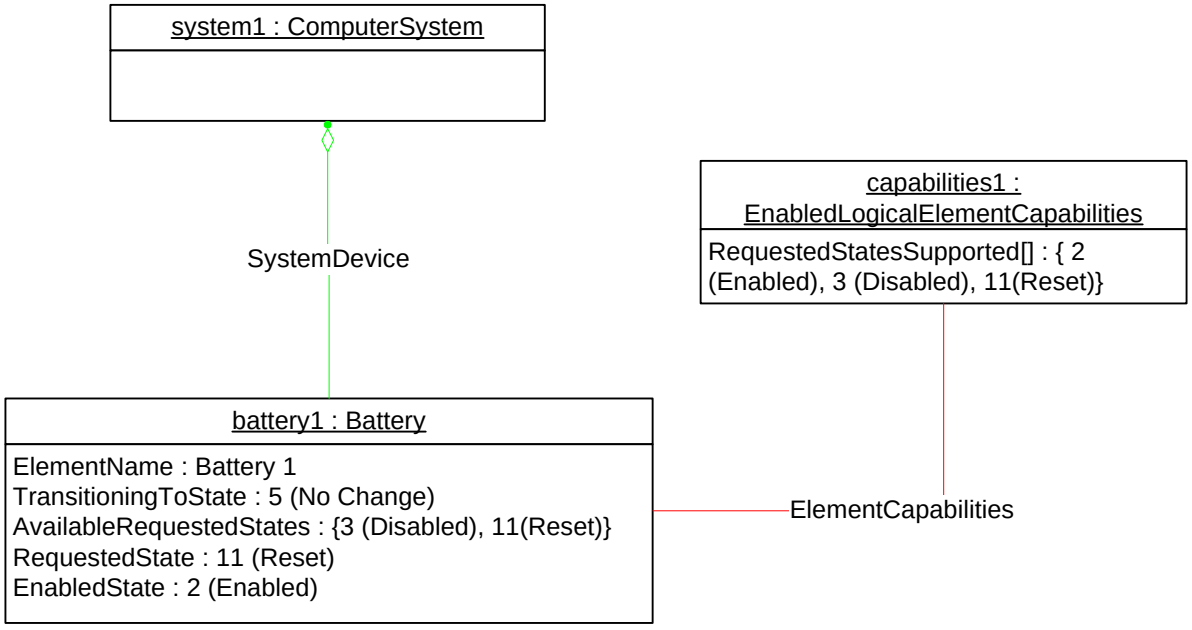

Figure 4 – Enabled Logical Element Profile: Transitioning Requested

9.2.2 Transitioning to Disabled State

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

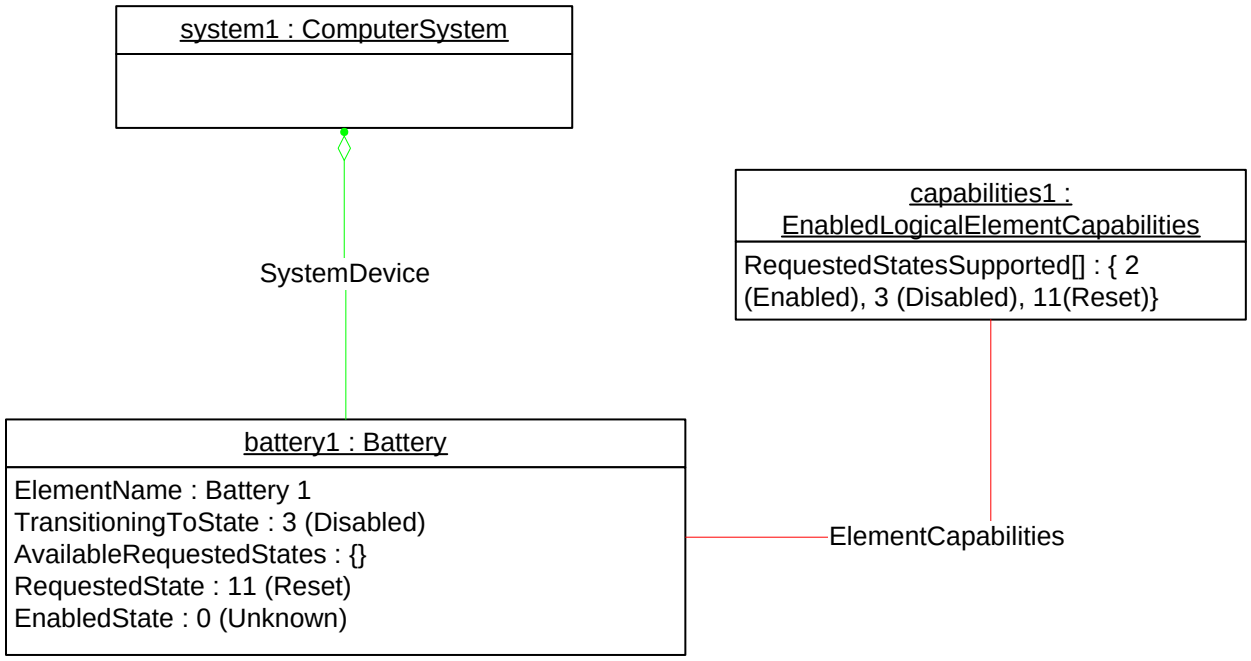

Figure 5 – Enabled Logical Element Profile: Transitioning to Disabled State

9.2.3 Transitioned to Disabled State

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

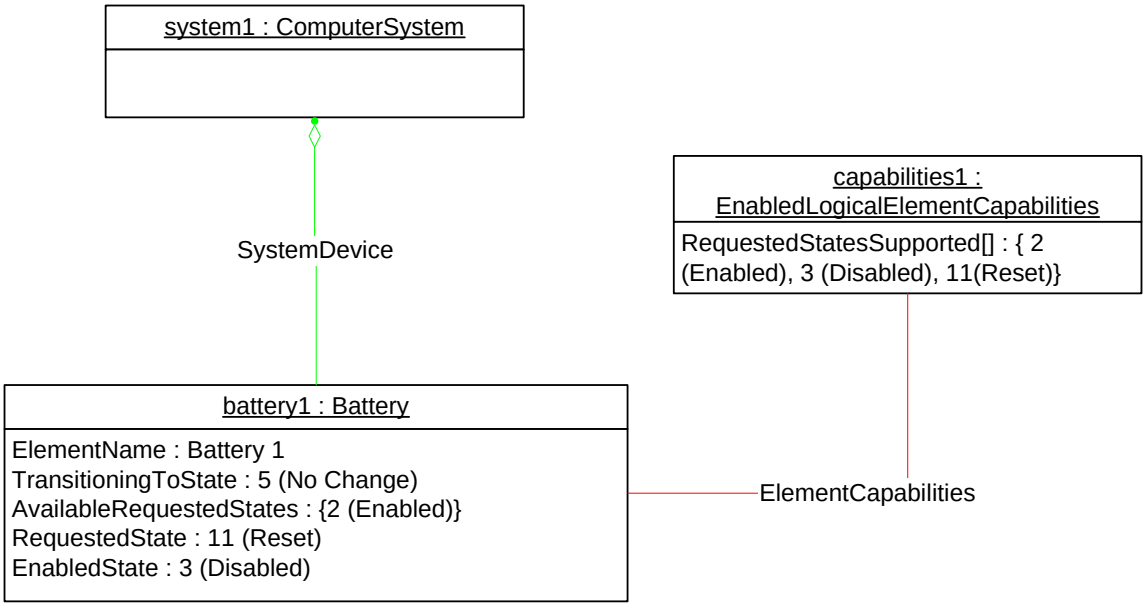

Figure 6 – Enabled Logical Element Profile: Disabled State

9.2.4 Transitioning to Enabled State

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

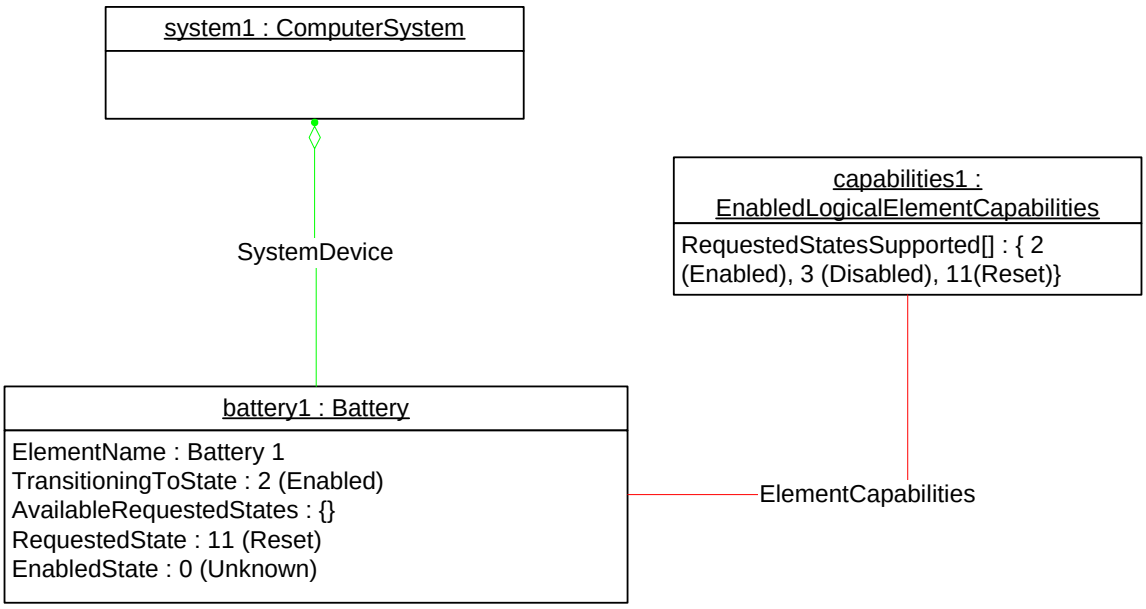

Figure 7 – Enabled Logical Element Profile: Transitioning to Enabled State

9.2.5 Transitioned to Enabled

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

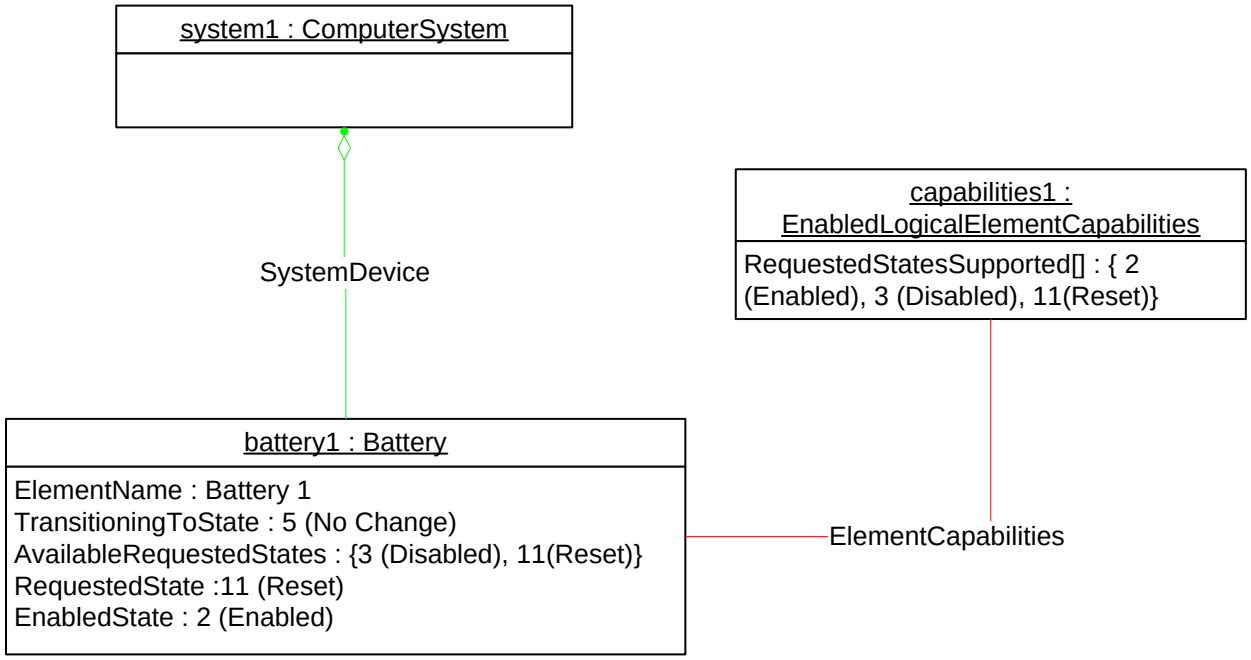

Figure 8 – Enabled Logical Element Profile: Transitioned to Enabled State

9.3 Determine the Level of State Management Supported

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

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

9.4 Enable the Enabled Logical Element

A client can enable the enabled logical element as follows:

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

9.5 Disable the Enabled Logical Element

A client can disable the enabled logical element as follows:

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

9.6 Reset the Enabled Logical Element

A client can reset the enabled logical element as follows:

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

9.7 Determine Whether the CIM_EnabledLogicalElement.ElementName Is Modifiable

A client can determine whether it can modify the CIM_EnabledLogicalElement.ElementName property as follows:

1) Find the CIM_EnabledLogicalElementCapabilities instance that is associated with the ELE instance.
2) Query the value of the ElementNameEditSupported property of the instance. If the value is TRUE, the client can modify the CIM_EnabledLogicalElement.ElementName property.

10 CIM Elements

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

Table 6 – CIM Elements: Enabled Logical Element Profile

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

10.1 CIM_ElementCapabilities

CIM_ElementCapabilities is used to associate an ELE instance with an instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the ELE instance. CIM_ElementCapabilities is mandatory if the CIM_EnabledLogicalElementCapabilities instance is instantiated.

Table 7 – CIM_ElementCapabilities

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

10.2 CIM_EnabledLogicalElementCapabilities

CIM_EnabledLogicalElementCapabilities represents the capabilities of the enabled logical element.

Table 8 – CIM_EnabledLogicalElementCapabilities

Properties	Requirement	Notes
InstanceID	Mandatory	**Key**
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.

10.3 CIM_EnabledLogicalElement

CIM_EnabledLogicalElement is an abstract class that is used to represent any enabled logical element.

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.

ANNEX A
(informative)

Change Log

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