Document Number: DSP1018

Date: 2013-03-28

Version: 1.1.1

Service Processor Profile

Document Type: Specification

Document Status: DMTF Standard

Document Language: en-US

CONTENTS

Figures

Tables

Foreword

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

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:

- Aaron Merkin – IBM
- Jeff Hilland - HP

Contributors:

- Jon Hass – Dell
- Khachatur Papanyan – Dell
- Enoch Suen – Dell
- Jeff Hilland – HP
- Christina Shaw – HP
- Aaron Merkin – IBM
- Perry Vincent – Intel
- John Leung – Intel
- Hemal Shah – Broadcom
- Satheesh Thomas - AMI

Introduction

The information in this specification 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 a service processor that is modeled using the DMTF Common Information Model (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.

168

Service Processor Profile

1 Scope

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

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/sites/default/files/standards/documents/DSP0200_1.1.0.html

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

DMTF DSP1075*, PCI Device Profile 1.0,*
http://www.dmtf.org/standards/published_documents/DSP1075_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 terms and definitions in DSP1033 and DSP1001 and the following 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 to conform to the document when the specified conditions are met

3.4
mandatory
indicates requirements to be followed strictly 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 "Referenced Profiles" table

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

3.10
shall not
indicates requirements to be followed strictly 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
unspecified
indicates that this profile does not define any constraints for the referenced CIM element or operation

3.14
service processor
a specialized device dedicated to management

3.15
standby service processor
an instance of CIM_ComputerSystem that represents a standby service processor of a redundancy set

4 Symbols and Abbreviated Terms

None.

5 Synopsis

Profile Name: Service Processor

Version: 1.1.1

Organization: DMTF

CIM Schema Version: 2.20

Central Class: CIM_ComputerSystem

Scoping Class: CIM_ComputerSystem

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

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.

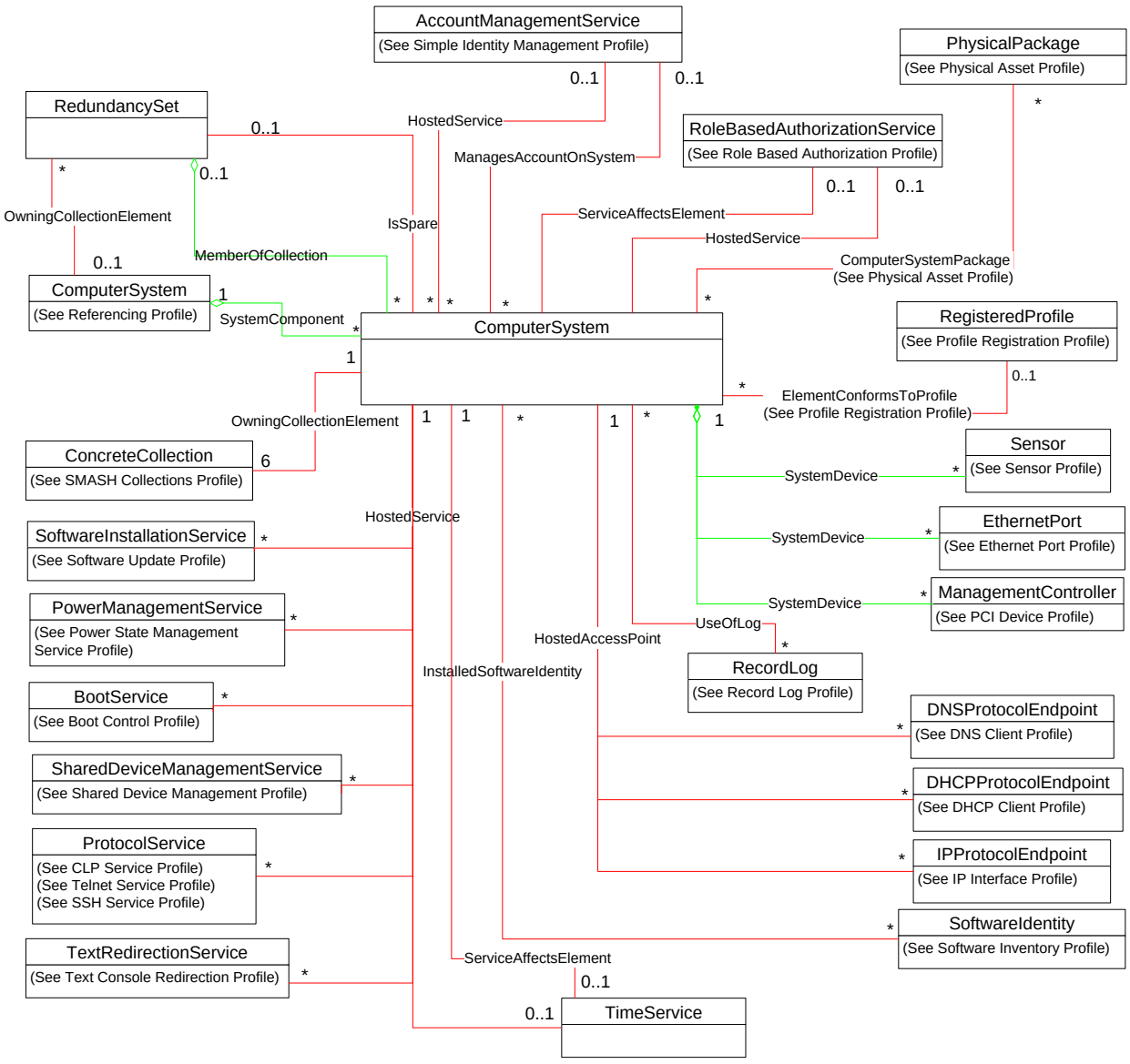
6 Description

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

Some examples of the service processors are:

- management processor (MP)
- service processor (SP)
- baseboard management controller (BMC)
- chassis manager

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



309

310 **Figure 1 – Service Processor Profile: Class Diagram**

311 7 Implementation

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

315 7.1 Representing a Service Processor

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

7.1.1 CIM_ComputerSystem.EnabledState

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

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).

7.1.2 Service Processor State Management Is Supported — Conditional

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

7.1.2.1 CIM_EnabledLogicalElementCapabilities

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

7.1.2.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported

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

7.1.2.2 CIM_ComputerSystem.RequestedState

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

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

7.1.2.3 CIM_ComputerSystem.EnabledState

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

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

7.1.3 Service Processor State Management Is Not Supported

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

7.1.3.1 CIM_EnabledLogicalElementCapabilities

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

7.1.3.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported

The CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property shall not contain any values.

7.1.3.2 CIM_ComputerSystem.RequestedState

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

7.1.4 Modifying ElementName Is Supported — Conditional

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

7.1.4.1 CIM_EnabledLogicalElementCapabilities

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

7.1.4.1.1 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported

The ElementNameEditSupported property shall have a value of TRUE.

7.1.4.1.2 CIM_EnabledLogicalElement.MaxElementNameLen

The MaxElementNameLen property shall be implemented.

7.1.5 Modifying ElementName Is Not Supported

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

7.1.5.1 CIM_EnabledLogicalElementCapabilities

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

7.1.5.1.1 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported

The ElementNameEditSupported shall have a value of FALSE.

7.1.5.1.2 CIM_EnabledLogicalElement.MaxElementNameLen

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

7.1.6 Representing the Physical Packaging (Optional)

Support for representing the physical packaging of the service processor is optional. The physical packaging may be modeled using one or more instances of CIM_PhysicalElement in accordance with the *Physical Asset Profile*.

7.2 Modeling Service Processor Redundancy (Optional)

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

At least one instance of CIM_RedundancySet shall exist.

7.2.1 Relationship between Redundancy Set and Redundant Service Processors

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

7.2.2 Relationship between Redundancy Set and Containing System

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

7.2.3 Active / Active Redundancy

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

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

7.2.4 Active / Standby Redundancy

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

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

- When the CIM_ComputerSystem.EnabledState property has the value 6 (Enabled but Offline), the SpareStatus property of the referencing CIM_IsSpare instance shall have the value 2 (Hot Standby).
- When the CIM_ComputerSystem.EnabledState property has the value 3 (Disabled), the SpareStatus property of the referencing CIM_IsSpare instance shall have the value 3 (Cold Standby).
- When the CIM_ComputerSystem.EnabledState property has a value other than 3 (Disabled) or 6 (Enabled but Offline), the SpareStatus property of the referencing CIM_IsSpare instance shall have the value 0 (Unknown).

7.3 Managing Service Processor Time (Optional)

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

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

7.4 User Account Management (Optional)

The *Simple Identity Management Profile* and the *Role Based Authorization Profile* may be implemented to model user access to the service processor. When the *Simple Identity Management Profile* is implemented, an instance of CIM_AccountManagementService shall be associated with the Central Instance through the CIM_HostedService association. When the *Role Based Authorization Profile* is implemented, an instance of CIM_RoleBasedAuthorizationService shall be associated with the Central Instance through the CIM_HostedService association.

7.5 Boot Control Profile (Optional)

The *Boot Control Profile* may be implemented to model the ability of the service processor to manage its own boot configuration or that of the systems it managed. If the *Boot Control Profile* is implemented, an instance of CIM_BootService shall be associated with the Central Instance through the CIM_HostedService association.

7.6 CLP Service Profile (Optional)

The *CLP Service Profile* may be implemented to model a CLP service hosted on the service processor. When the *CLP Service Profile* is implemented, at least one instance of CIM_ProtocolService shall be associated with the Central Instance through an instance of CIM_HostedService.

7.7 DHCP Client Profile (Optional)

The *DHCP Client Profile* may be implemented to model the DHCP client of a service processor. When the *DHCP Client Profile* is implemented, at least one instance of CIM_DHCPProtocolEndpoint shall be associated with the Central Instance through an instance of CIM_HostedAccessPoint.

7.8 DNS Client Profile (Optional)

The *DNS Client Profile* may be implemented to model the DNS client of a service processor. When the *DNS Client Profile* is implemented, at least one instance of CIM_DNSProtocolEndpoint shall be associated with the Central Instance through an instance of CIM_HostedAccessPoint.

7.9 Ethernet Port Profile (Optional)

The *Ethernet Port Profile* may be implemented to model an Ethernet interface of a service processor. When the *Ethernet Port Profile* is implemented, at least one instance of CIM_EthernetPort shall be associated with the Central Instance through an instance of CIM_SystemDevice.

7.10 Software Inventory Profile (Optional)

The *Software Inventory Profile* may be implemented to model the software version information of the service processor. When the *Software Inventory Profile* is implemented, at least one instance of CIM_SoftwareIdentity shall be associated with the Central Instance of this profile through an instance of CIM_InstalledSoftwareIdentity.

7.11 Software Update Profile (Optional)

The *Software Update Profile* may be implemented to model the ability of the service processor to update software installed on one or more components of managed systems, including the service processor itself. When the *Software Update Profile* is implemented, an instance of CIM_SoftwareInstallationService shall be associated with the Central Instance through an instance of CIM_HostedService.

7.12 IP Interface Profile (Optional)

The *IP Interface Profile* may be implemented to model the IP interface of a service processor. When the *IP Interface Profile* is implemented, at least one instance of CIM_IPProtocolEndpoint shall be associated with the Central Instance through an instance of CIM_HostedAccessPoint.

7.13 Physical Asset Profile (Optional)

The *Physical Asset Profile* may be implemented to model the physical package and physical asset information of a service processor. When the *Physical Asset Profile* is implemented, at least one instance of CIM_PhysicalPackage shall be associated with the Central Instance through an instance of CIM_ComputerSystemPackage.

7.14 Record Log Profile (Optional)

The *Record Log Profile* may be implemented to model one or more logs of the service processor. When the *Record Log Profile* is implemented, an instance of CIM_RecordLog shall be associated with Central Instance through an instance of CIM_UseOfLog.

7.15 Sensors Profile (Optional)

The *Sensors Profile* may be implemented to model the sensors of the service processor. When the *Sensors Profile* is implemented, at least one instance of CIM_Sensor or CIM_NumericSensor shall be associated with the Central Instance through an instance of CIM_SystemDevice.

7.16 Power State Management Profile (Optional)

The *Power State Management Profile* may be implemented to model the ability of the service processor to perform power control operations for the managed system or the service processor itself. When the *Power State Management Profile* is implemented, an instance of CIM_PowerManagementService shall be associated with the Central Instance through an instance of CIM_HostedService.

7.17 Shared Device Management Profile (Optional)

The *Shared Device Management Profile* may be implemented to model the ability of the service processor to control shared devices of a modular system. When the *Shared Device Management Profile* is implemented, an instance of CIM_SharedDeviceManagementService shall be associated with the Central Instance through an instance of CIM_HostedService.

7.18 SMASH Collections Profile (Optional)

The *SMASH Collections Profile* may be implemented. When the *SMASH Collections Profile* is implemented, each instance of CIM_ConcreteCollection that is defined by the *SMASH Collections Profile* shall be associated with the Central Instance through an instance of CIM_OwningCollectionElement.

7.19 SSH Service Profile (Optional)

The *SSH Service Profile* may be implemented to model an SSH service hosted on the service processor. When the *SSH Service Profile* is implemented, at least one instance of CIM_ProtocolService shall be

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

7.20 Telnet Service Profile (Optional)

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

7.21 Text Console Redirection Profile (Optional)

The *Text Console Redirection Profile* may be implemented to model the ability of the service processor to provide text console redirection for managed systems. When the *Text Console Redirection Profile* is implemented, at least one instance of CIM_TextRedirectionService shall be associated with the Central Instance through an instance of CIM_HostedService.

7.22 PCI Device Profile (Optional)

The *PCI Device Profile* may be implemented to model the ability of the service processor to provide PCI configuration information for managed systems. When the *PCI Device Profile* is implemented and the ServiceProcessor is modeled as a PCI device, at least one instance of CIM_ManagementController shall be associated with the Central Instance of this profile through an instance of CIM_SystemDevice and the CIM_ManagementController shall be associated with at least one instance of CIM_PCIDevice through an instance of CIM_ConcreteIdentity.

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_ComputerSystem.RequestStateChange( )

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

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

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

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

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

No standard messages are defined for this method.

542 **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.

543 **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

544 ### 8.1.1 RequestStateChange( ) for the Standby Service Processor

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

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

551 ## 8.2 Method: CIM_RedundancySet.Failover( )

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

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

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

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

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

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

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

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

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

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.

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

8.3 Method: CIM_TimeService.ManageTime( )

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

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

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.

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

8.4 Profile Conventions for Operations

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

The default list of operations is as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

8.5 CIM_ComputerSystem

Table 9 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 9, all operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

Table 9 – Operations: CIM_ComputerSystem

Operation	Requirement	Messages
ModifyInstance	Optional. See 8.5.1.	None

8.5.1 CIM_ComputerSystem — ModifyInstance

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

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

8.5.1.1 CIM_ComputerSystem.ElementName

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

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

8.6 CIM_HostedService

Table 10 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 10, all operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

Table 10 – Operations: CIM_HostedService

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

8.7 CIM_IsSpare

Table 11 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 11, all operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

Table 11 – Operations: CIM_IsSpare

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

8.8 CIM_ElementCapabilities

Table 12 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 12, all operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

Table 12 – Operations: CIM_ElementCapabilities

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

8.9 CIM_EnabledLogicalElementCapabilities

All operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

8.10 CIM_MemberOfCollection

Table 13 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 13, all operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

Table 13 – Operations: CIM_MemberOfCollection

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

8.11 CIM_RedundancySet

All operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

8.12 CIM_TimeService

All operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

8.13 CIM_ServiceAffectsElement

Table 14 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 14, all operations in the default list in 8.4 shall be implemented as defined in DSP0200.

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

Table 14 – Operations: CIM_ServiceAffectsElement

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

9 Use Cases

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

9.1 Object Diagrams

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

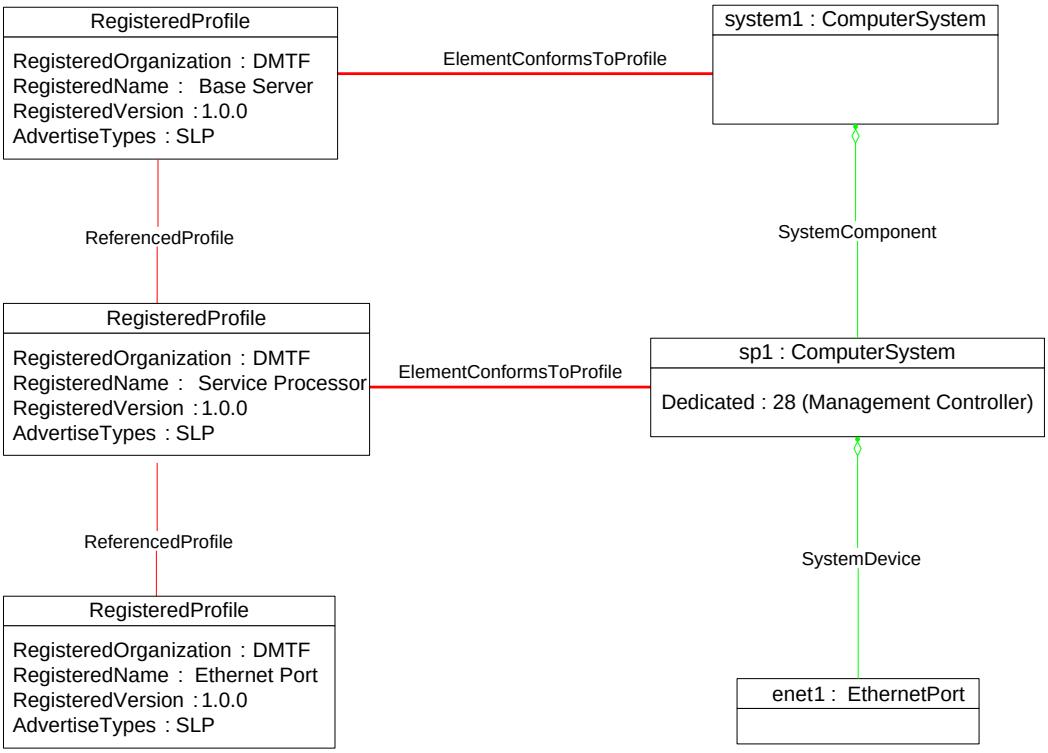


Figure 2 – Base Server

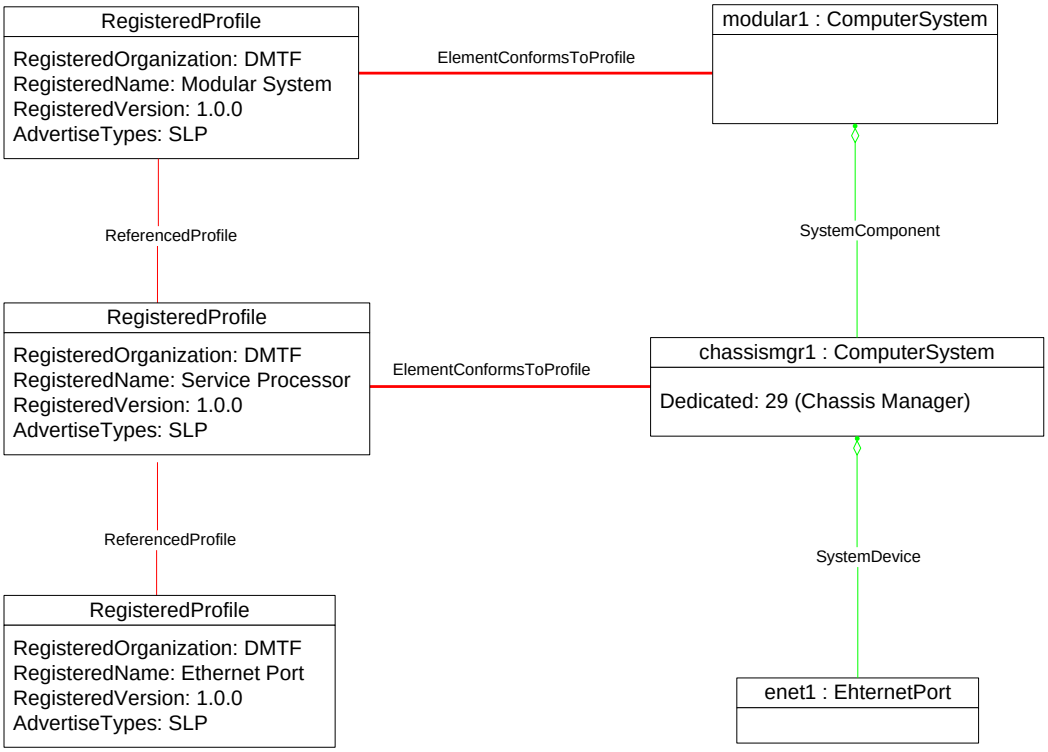


Figure 3 – Modular System

Figure 4 is an object diagram showing redundant service processors installed in a modular system. chassismgr1 is the active service processor. chassismgr2 is the backup service processor. This is indicated by the values of the EnabledState and RequestedState properties of the two instances and by the CIM_IsSpare association between the CIM_RedundancySet instance and chassismgr2.

In the illustrated system, a single configuration exists for the service processors. All functionality, including management interfaces, is hosted on and accessed at the active service processor. This is indicated by the active IP interface (ip1) bound to the Ethernet interface (enet2) of chassismgr1 and by the services (bootsvc1 and sharesvc1) associated through CIM_HostedService with chassismgr1.

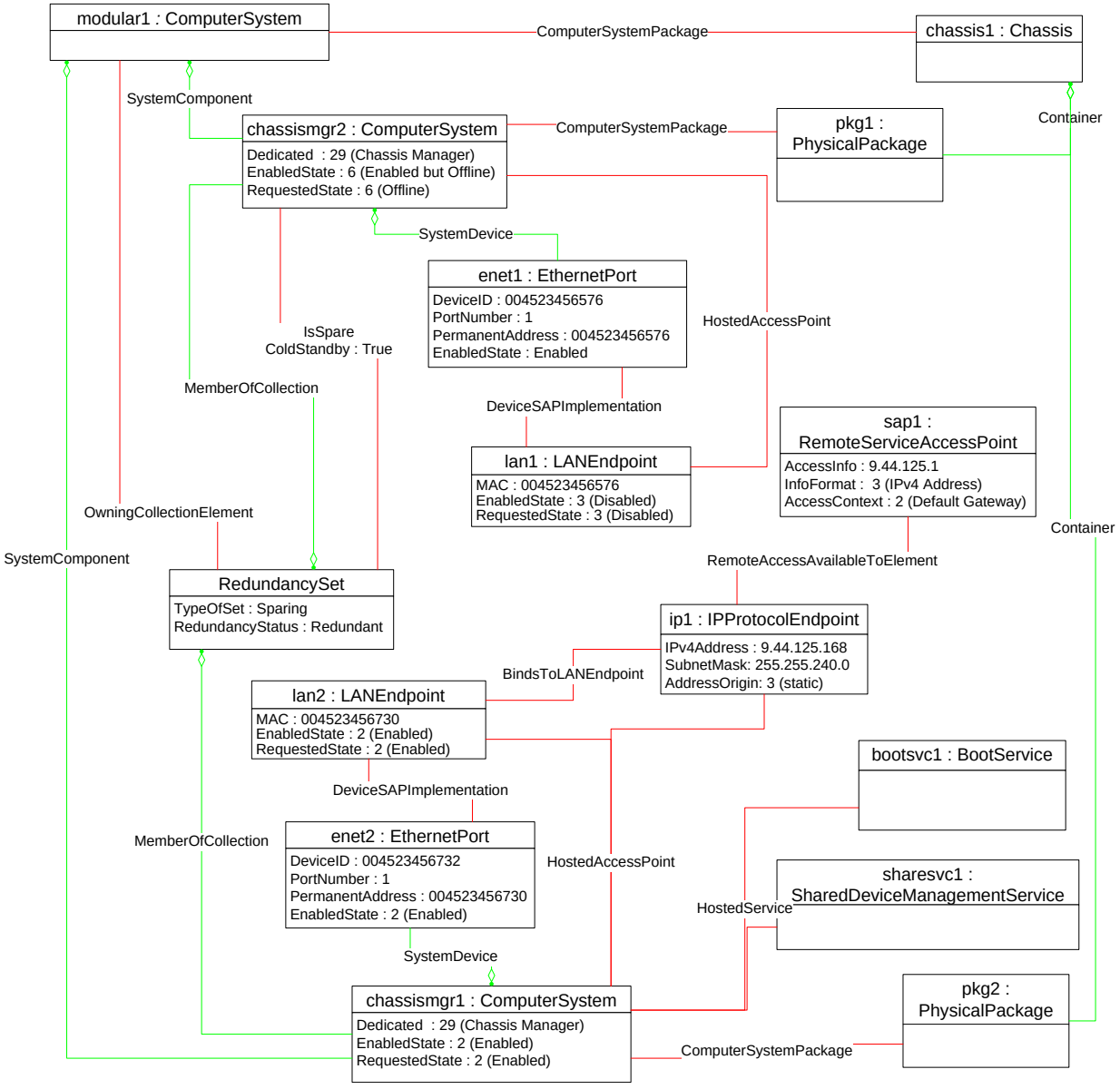


Figure 4 – Service Processors before Failover

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

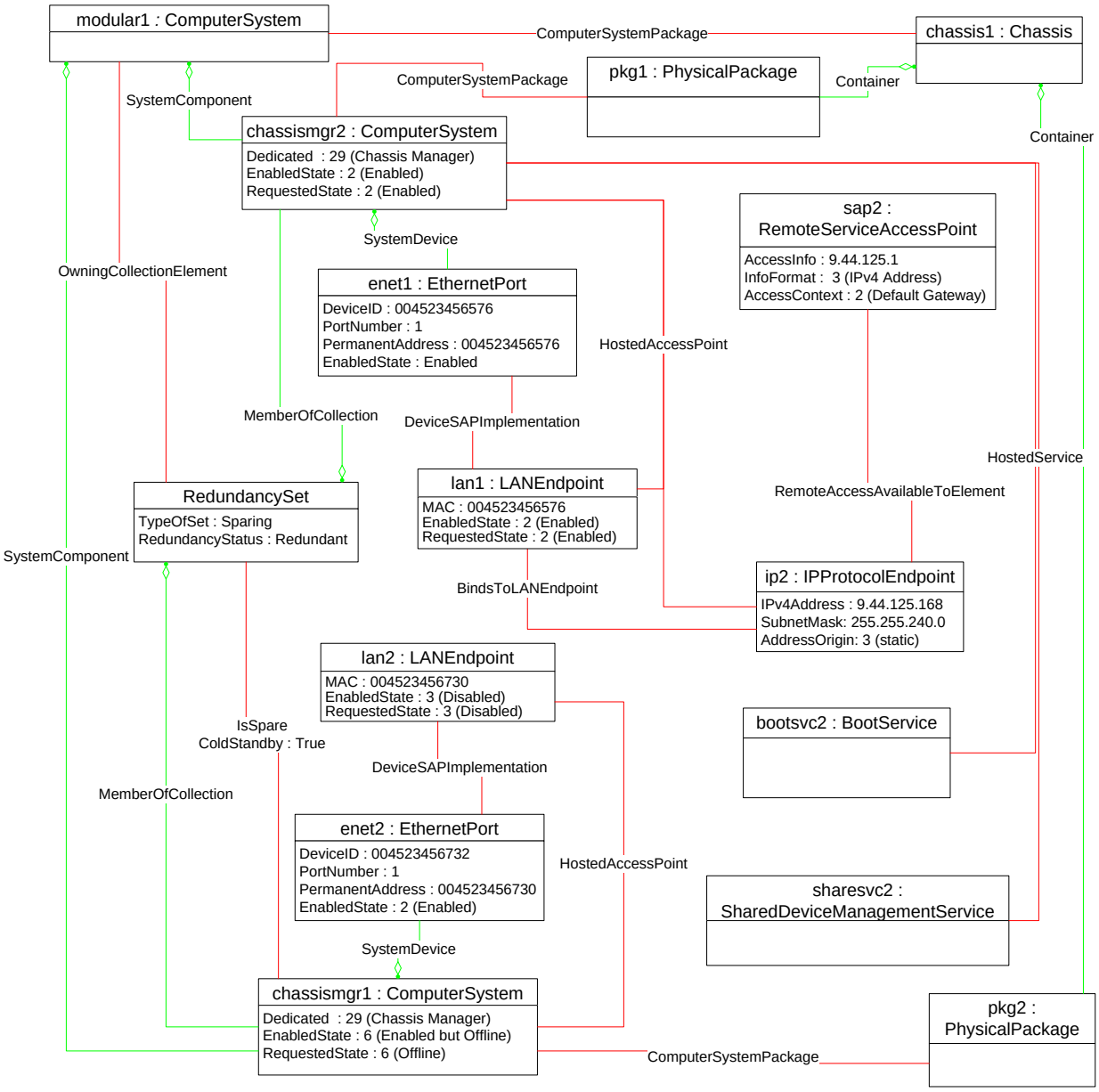


Figure 5 – Service Processors after Failover

9.2 Reset a Service Processor

A client can reset the service processor as follows:

1) For the given instance of CIM_ComputerSystem, find the associated instance of CIM_EnabledLogicalElementCapabilities.

2) If the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is a non-empty array that contains the value 11 (Reset), execute the RequestStateChange( ) method with the value of the RequestedState parameter set to 11 (Reset).

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

9.3 Retrieve the Service Processor Redundancy Status

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

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

9.4 Determine Whether Manual Failover Is Supported

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

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

9.5 Force a Service Processor Failover

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

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

9.6 Determine Whether the ElementName Is Modifiable

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

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

9.7 Determining If State Management Is Supported

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

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

10 CIM Elements

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

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		

10.1 CIM_ComputerSystem

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

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.

Elements	Requirement	Notes

10.2 CIM_ElementCapabilities

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

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

10.3 CIM_EnabledLogicalElementCapabilities

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

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.

10.4 CIM_HostedService

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

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

10.5 CIM_IsSpare

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

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.

10.6 CIM_MemberOfCollection

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

Table 21 – Class: CIM_MemberOfCollection

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

10.7 CIM_OwningCollectionElement

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

Table 22 – Class: CIM_OwningCollectionElement

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

10.8 CIM_RedundancySet

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

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 processor s 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.

10.9 CIM_RegisteredProfile

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

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 "1.1.1".

10.10 CIM_ServiceAffectsElement

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

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)

10.11 CIM_TimeService

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

Table 26 – Class: CIM_TimeService

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

10.12 CIM_ManagementController

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

Table 27 – Class: CIM_ManagementController

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

ANNEX A
(informative)

Change Log

Version	Date	Description
1.0.0	2009-06-22	
1.1.0	2011-06-30	Added support for the PCI Profile
1.1.1	2013-03-28	Removed HostedAccessPoint for RemoteServiceAccessPoint in Fig 4 and Fig 5 Renamed one of enet2 as enet1 in Fig 4