Profile Error: This profile contains 3 errors (search for 'Error:')

Document Number: XMP1009

Date: 2013-08-01

Version: 1.0.3m

Example Sensors Profile

IMPORTANT: This specification is not a standard. It does not necessarily reflect the views of the DMTF or all of its members. Because this document is a Work in Progress, this specification may still change, perhaps profoundly. This document is available for public review and comment until the stated expiration date.

This document expires on: **2014-01-31**.

Target version for DMTF Standard: **1.0.3**.

Provide any comments through the DMTF Feedback Portal: http://www.dmtf.org/standards/feedback

Document Type: Specification

Document Status: Work in Progress

Document Language: en-US

Copyright notice

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

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patent owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified after publication, and shall be indemnified and held harmless by any party implementing the standard from any and all claims of infringement by a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

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

19

CONTENTS

Foreword ... 5
Introduction ... 6
1 Scope ... 7
2 Normative references ... 7
3 Terms and definitions ... 7
3.1 General ... 7
4 Symbols and abbreviated terms ... 8
5 Synopsis ... 8
6 Description ... 10
7 Implementation ... 11
7.1 Features ... 11
7.1.1 Feature: SensorElementNameModification ... 11
7.1.2 Feature: SensorStateManagement ... 12
7.1.3 Feature: LowerThresholdNonCriticalSupported ... 12
7.1.4 Feature: UpperThresholdNonCriticalSupported ... 12
7.1.5 Feature: LowerThresholdCriticalSupported ... 13
7.1.6 Feature: UpperThresholdCriticalSupported ... 13
7.1.7 Feature: LowerThresholdFatalSupported ... 13
7.1.8 Feature: UpperThresholdFatalSupported ... 13
7.1.9 Feature: LowerThresholdNonCriticalSettable ... 14
7.1.10 Feature: UpperThresholdNonCriticalSettable ... 14
7.1.11 Feature: LowerThresholdCriticalSettable ... 14
7.1.12 Feature: UpperThresholdCriticalSettable ... 15
7.1.13 Feature: LowerThresholdFatalSettable ... 15
7.1.14 Feature: UpperThresholdFatalSettable ... 15
7.2 Adaptations ... 16
7.2.1 Conventions ... 16
7.2.2 Adaptation: ComputerSystem: CIM_ComputerSystem ... 16
7.2.3 Adaptation: SystemDevice: CIM_SystemDevice ... 17
7.2.4 Adaptation: AbstractSensor: CIM_Sensor ... 17
7.2.5 Adaptation: DiscreteSensor: CIM_Sensor ... 21
7.2.6 Adaptation: NumericSensor: CIM_NumericSensor ... 23
7.2.7 Adaptation: SensorCapabilities: CIM_EnabledLogicalElementCapabilities ... 27
7.2.8 Adaptation: ElementCapabilities: CIM_ElementCapabilities ... 28
7.2.9 Adaptation: SensoredElement: CIM_ManagedSystemElement ... 29
7.2.10 Adaptation: AssociatedSensor: CIM_AssociatedSensor ... 30
8 Use cases and state descriptions ... 30
8.1 State description: SimpleObjectDiagram ... 30
8.2 Use case: ShowAllSensorStates ... 31
8.3 Use case: FindSensorsOfSystemElement ... 32

8.4 Use case: ChangeThreshold 32
8.5 Use case: ResetThresholds 32
8.6 Use case: DetermineElementNameModifiability 32
8.7 Use case: DetermineStateManagementSupport 33
ANNEX A (informative) Change log 34
Bibliography 35

20

Figures

Figure 1 – DMTF collaboration structure diagram for the Sensors profile 10
Figure 2 – DMTF collaboration structure diagram for the ConcreteSensors component of the Sensors profile 11
Figure 3 – Simple object diagram 31

21

Tables

Table 1 – Profile references 9
Table 2 – Features 9
Table 3 – Adaptations 9
Table 4 – Use cases and state descriptions 10
Table 5 – ComputerSystem: Element requirements 16
Table 6 – SystemDevice: Element requirements 17
Table 7 – AbstractSensor: Element requirements 18
Table 8 – Mapping between EnabledState values and sensor states 19
Table 9 – RequestStateChange( ): Parameter requirements 20
Table 10 – RequestedState: Value requirements 20
Table 11 – RequestStateChange: Return values 21
Table 12 – DiscreteSensor: Element requirements 21
Table 13 – NumericSensor: Element requirements 23
Table 14 – RestoreDefaultThresholds( ): Parameter requirements 26
Table 15 – RestoreDefaultThresholds: Return values 26
Table 16 – SensorCapabilities: Element requirements 27
Table 17 – ElementCapabilities: Element requirements 29
Table 18 – SensoredElement: Element requirements 29
Table 19 – AssociatedSensor: Element requirements 30

22

Foreword

This document was prepared by the DMTF Physical Platform Profiles Working Group and Server Management Working Group

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

Design Note: This document contains design notes (like this one), that provide information about the way the document is written, or to demonstrate certain things. Such design notes would not appear in a released version of this document.

Design Note: This MRP document represents DSP1009 1.0.2 as a machine readable profile in MRP 1.0 format. The MRP 1.1 build environment does support MRP 1.0 profiles. Since machine readable profiles need to be compliant to DSP1001 1.1, this document utilizes the newly introduced concepts, such as adaptations, features and collaboration diagrams. Its HTML version generated by the DSP8029 XSL stylesheet uses the new condensed format defined in DSP1001 1.1.

Design Note: This profile demonstrates how boilerplate text can be replaced with references to messages defined in an MRP Organization Message Registry. The copyright text demonstrates how to specify dynamic values within the message, for the copyright years.

Acknowledgements

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

- Jon Hass, Dell Inc. (editor)
- Khachatur Papanyan, Dell Inc. (editor)
- Jim Davis, WBEM Solutions (editor)
- Linda Martinez, Dell Inc.
- Jeff Hilland, HP
- Christina Shaw, HP
- Aaron Merkin, IBM
- Jeff Lynch, IBM
- Perry Vincent, Intel
- John Leung, Intel

Introduction

40 This document defines the usage of CIM classes used to represent and manage sensors in a managed environment. The document also defines the usage of CIM association classes that describe the relationship of the sensors with the sensored elements they measure and with DMTF profile implementation information. The information in this document is intended to be sufficient for a provider or consumer of this data to identify unambiguously the classes, properties, methods, and values that need to be instantiated and manipulated.

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

42

Document conventions

43

Typographical conventions

44 The following typographical conventions are used in this document:

45 • Document titles are marked in *italics*.

46 • Important terms that are used for the first time are marked in *italics*.

47 • Terms include a link to the term definition in the "Terms and definitions" clause, enabling easy navigation to the term definition.

48

OCL usage conventions

49 Constraints in this document are specified using OCL (see OCL 2.0).

50 OCL statements are in `monospaced font`.

51

Example Sensors Profile

52

1 Scope

53 The Sensors profile extends the management capabilities of referencing profiles by adding the capability to represent sensors , their relationship with the sensored elements they measure, and the implementation of this profile.

54

2 Normative references

55 The following referenced documents are indispensable for the application of this document. For dated or versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies. For references without a date or version, the latest published edition of the referenced document (including any corrigenda or DMTF update versions) applies.

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

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

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

59 DMTF XMP1033, *Example Profile Registration Profile (sample profile in DSP2023) 1.0*, http://www.dmtf.org/standards/published_documents/DSP2023_1.0.zip

60 OMG formal/06-05-01, *Object Constraint Language 2.0*, http://www.omg.org/spec/OCL/2.0/

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

62

3 Terms and definitions

63 In this document, some terms have a specific meaning beyond the normal English meaning. Those terms are defined in this clause.

64

3.1 General

65 The terms "shall" ("required"), "shall not", "should" ("recommended"), "should not" ("not recommended"), "may", "need not" ("not required"), "can" and "cannot" in this document are to be interpreted as described in ISO/IEC Directives, Part2, Annex H. The terms in parenthesis are alternatives for the preceding term, for use in exceptional cases when the preceding term cannot be used for linguistic reasons. Note that ISO/IEC Directives, Part2, Annex H specifies additional alternatives. Occurrences of such additional alternatives shall be interpreted in their normal English meaning in this document.

66 The terms "clause", "subclause", "paragraph", "annex" in this document are to be interpreted as described in ISO/IEC Directives, Part2, Clause 5.

67 The terms "normative" and "informative" in this document are to be interpreted as described in ISO/IEC Directives, Part2, Clause 3. In this document, clauses, subclauses or annexes indicated with "(informative)" as well as notes and examples do not contain normative content.

The terms defined in DSP0004, DSP0223, and DSP1001 apply to this document.

68 The following additional terms are defined in this document.

69 **3.2**

70 **analog sensor**
sensor that measures an analog quantity (e.g. voltage) and provides a numeric value for the quantity. Also called a numeric sensor.

71 **3.3**

72 **discrete sensor**
sensor that provides discrete values for the quantity it measures (e.g. a lock is closed or open).

73 **3.4**

74 **sensor**
device that measures a physical quantity and makes a value for the quantity accessible to a programmatic observer.

75 **3.5**

76 **sensored element**
element in the managed environment that is measured by a sensor .

77

4 Symbols and abbreviated terms

78 The abbreviations defined in DSP0004, DSP0223, and DSP1001 apply to this document.

79 This document does not define any additional abbreviations.

80

5 Synopsis

81 **Profile name:** Example Sensors

82 **Version:** 1.0.3

83 **Organization:** DMTF

84 **Abstract:** No

85 **Profile type:** Component

86 **Schema:** DMTF CIM 2.22 (experimental)

87 **Central class adaptation:** AbstractSensor

88 **Scoping class adaptation:** ComputerSystem

89 **Scoping path:** SystemDevice

90 The Sensors profile extends the management capabilities of referencing profiles by adding the capability to represent, monitor and control sensors , and to represent their relationship with sensored elements .

91 Table 1 identifies the profile references defined in this profile.

92

Table 1 – Profile references

Profile reference name	Profile name	Organi-zation	Version	Relation-ship	Description
PRP	Profile Registration	DMTF	1.0	Mandatory	Used to represent the implementation of this profile.

93 Table 2 identifies the features defined in this profile.

94 **Table 2 – Features**

Feature	Requirement	Description
SensorElementNameModification	Optional	See 7.1.1.
SensorStateManagement	Optional	See 7.1.2.
LowerThresholdNonCriticalSupported	Optional	See 7.1.3.
UpperThresholdNonCriticalSupported	Optional	See 7.1.4.
LowerThresholdCriticalSupported	Optional	See 7.1.5.
UpperThresholdCriticalSupported	Optional	See 7.1.6.
LowerThresholdFatalSupported	Optional	See 7.1.7.
UpperThresholdFatalSupported	Optional	See 7.1.8.
LowerThresholdNonCriticalSettable	Optional	See 7.1.9.
UpperThresholdNonCriticalSettable	Optional	See 7.1.10.
LowerThresholdCriticalSettable	Optional	See 7.1.11.
UpperThresholdCriticalSettable	Optional	See 7.1.12.
LowerThresholdFatalSettable	Optional	See 7.1.13.
UpperThresholdFatalSettable	Optional	See 7.1.14.

95 Table 3 identifies the class adaptations defined in this profile.

96 **Table 3 – Adaptations**

Adaptation	Elements	Requirement	Description
Instantiated, embedded and abstract adaptations			
ComputerSystem	CIM_ComputerSystem	Mandatory	See 7.2.2.
SystemDevice	CIM_SystemDevice	Mandatory	See 7.2.3.
AbstractSensor	CIM_Sensor	See derived adaptations	See 7.2.4.
DiscreteSensor	CIM_Sensor	Mandatory	See 7.2.5.
NumericSensor	CIM_NumericSensor	Mandatory	See 7.2.6.
SensorCapabilities	CIM_EnabledLogicalElementCapabilities	Conditional	See 7.2.7.
ElementCapabilities	CIM_ElementCapabilities	Conditional	See 7.2.8.
SensoredElement	CIM_ManagedSystemElement	See derived adaptations	See 7.2.9.
AssociatedSensor	CIM_AssociatedSensor	Mandatory	See 7.2.10.
Indications and exceptions			
This profile does not define any such adaptations.			

97 Table 4 identifies the use cases and state descriptions defined in this profile.

98

Table 4 – Use cases and state descriptions

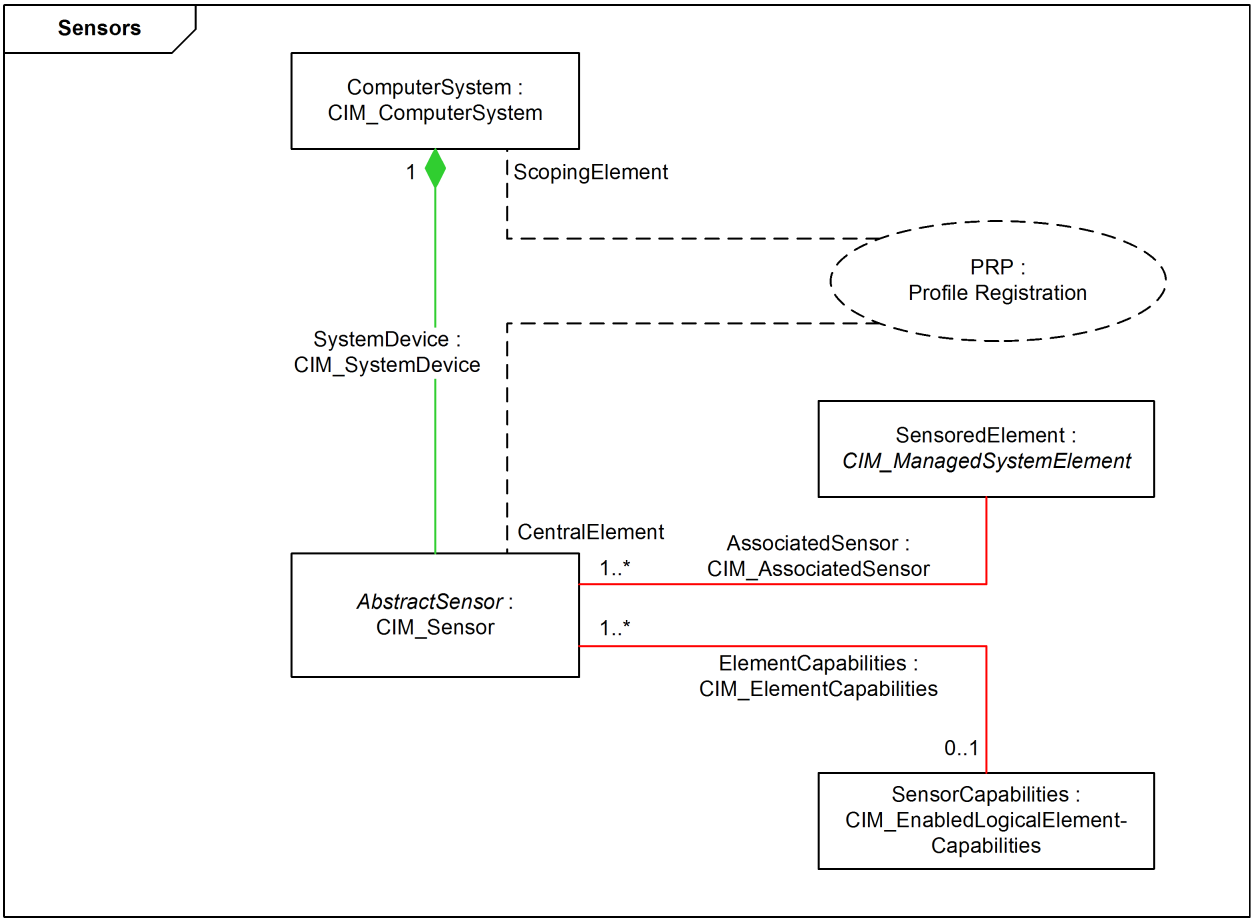
Name	Description
State description: SimpleObjectDiagram	See 8.1.
Use case: ShowAllSensorStates	See 8.2.
Use case: FindSensorsOfSystemElement	See 8.3.
Use case: ChangeThreshold	See 8.4.
Use case: ResetThresholds	See 8.5.
Use case: DetermineElementNameModifiability	See 8.6.
Use case: DetermineStateManagementSupport	See 8.7.

99

6 Description

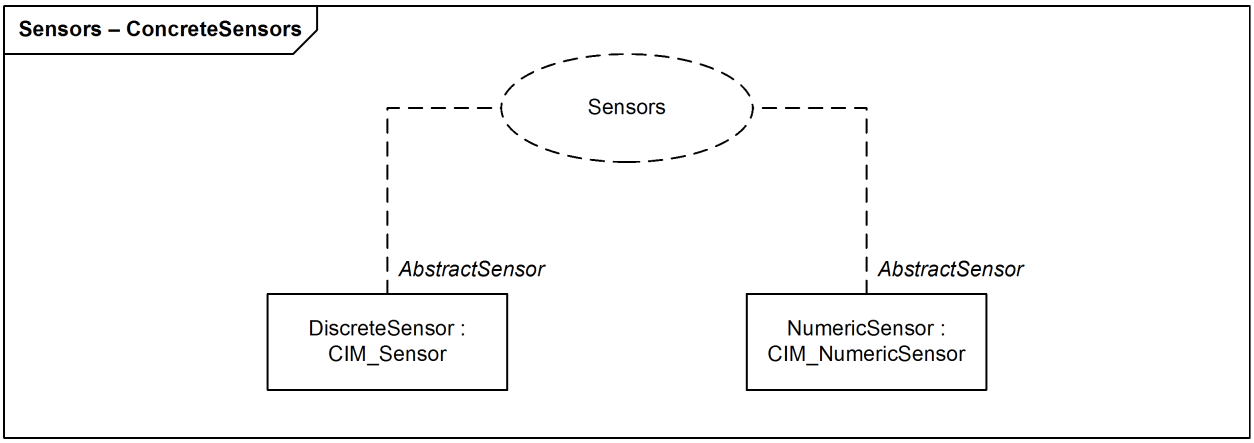
100 The following two DMTF collaboration structure diagrams show all class adaptations defined in this profile, and all profiles referenced by this profile.

101



102 **Figure 1 – DMTF collaboration structure diagram for the Sensors profile**

103



104 **Figure 2 – DMTF collaboration structure diagram for the ConcreteSensors component of the Sensors profile**

105

The logical aspect of sensors in the managed environment is represented by instances of the following adaptations that are based on the abstract base adaptation AbstractSensor :

106 • The DiscreteSensor adaptation, representing discrete sensors .

107 • The NumericSensor adaptation, representing analog sensors .

108 The system hosting a sensor is represented by a ComputerSystem instance associated via SystemDevice to the AbstractSensor instance representing the sensor .

109 The sensored elements of a sensor are represented by SensoredElement instances associated via AssociatedSensor to the AbstractSensor instance representing the sensor . If no such instance is associated to a AbstractSensor instance, the corresponding sensor's scope is the entire system that hosts the sensor.

110 The capabilities of a sensor and of the implementation of its AbstractSensor adaptation are represented by a SensorCapabilities instance associated via ElementCapabilities to the AbstractSensor instance representing the sensor . SensorCapabilities instances can be shared between multiple AbstractSensor instances.

111 Conformance of an implementation to this profile is represented through the PRP profile.

112

7 Implementation

113

7.1 Features

114

7.1.1 Feature: SensorElementNameModification

115 **Requirement level:** Optional

116 Implementing this feature provides support for client modification of the ElementName property of a AbstractSensor instance.

117 This feature can be made available to clients at the granularity of AbstractSensor instances.

118 It can be concluded that the feature is available for a AbstractSensor instance if:

120 • The following OCL derivation constraint evaluates to a Boolean value of True.

121 OCL context: A AbstractSensor instance.

122

```
derive: self.ElementCapabilities::Capabilities.ElementNameEditSupported
```

Otherwise, it can be concluded that the feature is not available.

7.1.2 Feature: SensorStateManagement

Requirement level: Optional

Implementing this feature provides support for client management of the state of a sensor by means of the RequestStateChange( ) method.

This feature can be made available to clients at the granularity of AbstractSensor instances.

It can be concluded that the feature is available for a AbstractSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A AbstractSensor instance.

```
derive: self.ElementCapabilities::Capabilities.
    RequestedStatesSupported->notEmpty()
```

Otherwise, it can be concluded that the feature is not available.

7.1.3 Feature: LowerThresholdNonCriticalSupported

Requirement level: Optional

Implementing this feature provides support for the LowerThresholdNonCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SupportedThresholds->
    select( entry | entry = 0 /* LowerThresholdNonCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.4 Feature: UpperThresholdNonCriticalSupported

Requirement level: Optional

Implementing this feature provides support for the UpperThresholdNonCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SupportedThresholds->
    select( entry | entry = 1 /* UpperThresholdNonCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.5 Feature: LowerThresholdCriticalSupported

Requirement level: Optional

Implementing this feature provides support for the LowerThresholdCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SupportedThresholds->
    select( entry | entry = 2 /* LowerThresholdCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.6 Feature: UpperThresholdCriticalSupported

Requirement level: Optional

Implementing this feature provides support for the UpperThresholdCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SupportedThresholds->
    select( entry | entry = 3 /* UpperThresholdCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.7 Feature: LowerThresholdFatalSupported

Requirement level: Optional

Implementing this feature provides support for the LowerThresholdFatal threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SupportedThresholds->
    select( entry | entry = 4 /* LowerThresholdFatal */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.8 Feature: UpperThresholdFatalSupported

Requirement level: Optional

Implementing this feature provides support for the UpperThresholdFatal threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SupportedThresholds->
    select( entry | entry = 5 /* UpperThresholdFatal */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.9 Feature: LowerThresholdNonCriticalSettable

Requirement level: Optional

Implementing this feature provides support for setting the LowerThresholdNonCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SettableThresholds->
    select( entry | entry = 0 /* LowerThresholdNonCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.10 Feature: UpperThresholdNonCriticalSettable

Requirement level: Optional

Implementing this feature provides support for setting the UpperThresholdNonCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SettableThresholds->
    select( entry | entry = 1 /* UpperThresholdNonCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.11 Feature: LowerThresholdCriticalSettable

Requirement level: Optional

Implementing this feature provides support for setting the LowerThresholdCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

OCL context: A NumericSensor instance.

```
derive: self.SettableThresholds->
    select( entry | entry = 2 /* LowerThresholdCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.12 Feature: UpperThresholdCriticalSettable

Requirement level: Optional

Implementing this feature provides support for setting the UpperThresholdCritical threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SettableThresholds->
    select( entry | entry = 3 /* UpperThresholdCritical */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.13 Feature: LowerThresholdFatalSettable

Requirement level: Optional

Implementing this feature provides support for setting the LowerThresholdFatal threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SettableThresholds->
    select( entry | entry = 4 /* LowerThresholdFatal */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.1.14 Feature: UpperThresholdFatalSettable

Requirement level: Optional

Implementing this feature provides support for setting the UpperThresholdFatal threshold of an analog sensor .

This feature can be made available to clients at the granularity of NumericSensor instances.

It can be concluded that the feature is available for a NumericSensor instance if:

- The following OCL derivation constraint evaluates to a Boolean value of True.

  OCL context: A NumericSensor instance.

```
derive: self.SettableThresholds->
    select( entry | entry = 5 /* UpperThresholdFatal */ )->size() = 1
```

Otherwise, it can be concluded that the feature is not available.

7.2 Adaptations

7.2.1 Conventions

This profile defines operation requirements based on DSP0223.

For adaptations of ordinary classes and of associations, the requirements for operations are defined in adaptation-specific subclauses of subclause 7.2.

For association traversal operation requirements that are specified only in the elements table of an adaptation (i.e., without operation-specific subclauses), the names of the association adaptations to be traversed are listed in the elements table.

The default initialization requirement level for property requirements is optional.

The default modification requirement level for property requirements is optional.

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

- In: indicates that the parameter is an input parameter
- Out: indicates that the parameter is an output parameter
- Key: indicates that the property is a key (that is, its value is part of the instance path)
- Required: indicates that the element value shall be non-Null
- Null OK: indicates explicitly that the element value may be Null for mandatory, conditional or conditional exclusive properties. This information is not specified as a qualifier in the schema but as an indicator in the profile.

7.2.2 Adaptation: ComputerSystem: CIM_ComputerSystem

This adaptation models systems hosting sensors .

Adaptation type: Ordinary class

Implementation type: Instantiated

Requirement level: Mandatory

Table 5 – ComputerSystem: Element requirements

Element	Requirement	Description
Operations		
Associators( )	Mandatory	
AssociatorNames( )	Mandatory	
References( )	Mandatory	
ReferenceNames( )	Mandatory	

7.2.3 Adaptation: SystemDevice: CIM_SystemDevice

7.2.3.1 General

Adaptation type: Association class

Implementation type: Instantiated

Requirement level: Mandatory

This adaptation models the relationship between sensors that are represented by AbstractSensor instances and the systems hosting these sensors that are represented by ComputerSystem instances.

Table 6 – SystemDevice: Element requirements

Element	Requirement	Description
Properties		
GroupComponent	Mandatory	Key, see 7.2.3.2
PartComponent	Mandatory	Key, see 7.2.3.3
Operations		
GetInstance( )	Mandatory	

7.2.3.2 Property: GroupComponent

Requirement level: Mandatory

Reference kind: REF-typed

Constraints:

- Referenced instances shall be of class adaptation ComputerSystem.
- The multiplicity of this association end is 1 .. 1

7.2.3.3 Property: PartComponent

Requirement level: Mandatory

Reference kind: REF-typed

Constraints:

- Referenced instances shall be of class adaptation AbstractSensor.
- The multiplicity of this association end is 0 .. *

7.2.4 Adaptation: AbstractSensor: CIM_Sensor

7.2.4.1 General

Adaptation type: Ordinary class

Implementation type: Abstract

Requirement level: Defined by its derived adaptations

This abstract adaptation provides a basis for derived adaptations in this or other profiles, that model specific types of sensors .

This profile defines the DiscreteSensor and NumericSensor adaptations to be based on this adaptation.

Design Note: This adaptation demonstrates several ways to define the associations traversed by association traversal operations. Note the different ways mrp:Association child elements are specified vs. defaulted. Note that if no associations are specified in the MRP profile, they are defaulted and appear explicitly in the HTML output. Note that a certain number of traversed associations triggers the generation of subclauses for the operations.

299 **Constraint:**

300 OCL constraint in the context of a AbstractSensor instance:

301
```
inv: ( self.mrpIsFeatureAvailable('SensorElementNameModification') or
    self.mrpIsFeatureAvailable('SensorStateManagement') )
implies
    self.ElementCapabilities::Capabilities->size() = 1
```

302

Table 7 – AbstractSensor: Element requirements

Element	Requirement	Description
Properties		
SystemCreationClassName	Mandatory	Key
SystemName	Mandatory	Key
CreationClassName	Mandatory	Key
DeviceID	Mandatory	Key
ElementName	Mandatory	See 7.2.4.2
SensorType	Mandatory	
OtherSensorTypeDescription	Conditional	See 7.2.4.3
EnabledState	Mandatory	See 7.2.4.4
RequestedState	Mandatory	See 7.2.4.5
OperationalStatus	Mandatory	
HealthState	Mandatory	
Methods		
RequestStateChange( )	Conditional	See 7.2.4.6
Operations		
GetInstance( )	Mandatory	
EnumerateInstances( )	Mandatory	
EnumerateInstanceNames( )	Mandatory	
ModifyInstance( )	Conditional	See 7.2.4.7
Associators( )	Mandatory	
AssociatorNames( )	Mandatory	
References( )	Mandatory	
ReferenceNames( )	Mandatory	

303

7.2.4.2 Property: ElementName

304 **Requirement level:** Mandatory

305 **Constraint:**

306 OCL constraint in the context of a AbstractSensor instance:

307

```
inv: self.ElementName.size() <=
self.ElementCapabilities::Capabilities.MaxElementNameLen
```

308 **Modification requirement:**

Conditional exclusive

309 **Condition:**

The SensorElementNameModification feature is implemented.

310 In order to enforce the length restriction specified in the MaxElementNameLen property of the associated SensorCapabilities instance, the implementation may reject or truncate property values that exceed that length.

311 #### 7.2.4.3 Property: OtherSensorTypeDescription

312 **Requirement level:** Conditional

313 **Condition:**

314 The following OCL statement evaluates to true in the context of a AbstractSensor instance:

315
```
derive: self.SensorType = 1 /* Other */
```

317 #### 7.2.4.4 Property: EnabledState

318 **Requirement level:** Mandatory

319 Table 8 describes the mapping between values of this property and the corresponding sensor states.

320 **Table 8 – Mapping between EnabledState values and sensor states**

Value	Value description	Sensor state
2	Enabled	The sensor shall be operational.
3	Disabled	The sensor shall be disabled.
5	Not Applicable	The sensor's state is indeterminate, or sensor state management is not supported.

321 The value of this property may change as a result of the execution of the RequestStateChange( ) method or a change to the sensor's state by a non-CIM implementation.

322 #### 7.2.4.5 Property: RequestedState

323 **Requirement level:** Mandatory

324 **Constraints:**

326 • OCL constraint in the context of a AbstractSensor instance:

327
```
inv: if self.mrpIsFeatureAvailable('SensorStateManagement')
then (
    self.RequestedState = 5 /* No Change */ or
    self.RequestedState = 12 /* Not Applicable */ or
    self.ElementCapabilities::Capabilities.
        RequestedStatesSupported->contains(self.RequestedState)
) else (
    self.RequestedState = 12 /* Not Applicable */
)
```

329 • OCL constraint in the context of a AbstractSensor instance:

330

```
init: if self.mrpIsFeatureAvailable('SensorStateManagement')
then (
    self.RequestedState = 5 /* No Change */
)
```

7.2.4.6 Method: RequestStateChange( )

Requirement level: Conditional

Condition:

The SensorStateManagement feature is implemented.

Successful execution of this method on an instance of this adaptation shall change the instance's state to the value specified in the RequestedState parameter.

Invoking this method multiple times may result in earlier requests being overwritten or lost.

Constraints:

- OCL constraint in the context of a AbstractSensor instance:

```
pre: self.ElementCapabilities::Capabilities.
    RequestedStatesSupported->contains(RequestedState)
```

- OCL constraint in the context of a AbstractSensor instance:

```
post: self.RequestedState = RequestedState
```

Table 9 – RequestStateChange( ): Parameter requirements

Parameter	Description
RequestedState	In, see 7.2.4.6.1
Job	Out, see 7.2.4.6.2
TimeoutPeriod	In, see 7.2.4.6.3
Return value	See 7.2.4.6.4

7.2.4.6.1 Parameter: RequestedState

Table 10 – RequestedState: Value requirements

Value	Requirement	Description
2	Mandatory	2 = Enabled; the sensor shall become operational.
3	Mandatory	3 = Disabled; the sensor shall become disabled.

7.2.4.6.2 Parameter: Job

If the request is being executed asynchronously, the value of this parameter shall reference a ConcreteJob**Profile Error: A class adaptation "ConcreteJob" is referenced in a class adaptation link but is not defined or is defined more than once in this profile.** instance representing the asynchronously executing request. Otherwise, the value of this parameter shall be NULL.

Constraint:

Referenced instances shall be of class adaptation ConcreteJob**Profile Error: A class adaptation "ConcreteJob" is referenced in a class adaptation link but is not defined or is defined more than once in this profile.**.

7.2.4.6.3 Parameter: TimeoutPeriod

Client-specified maximum amount of time the transition to a new state is supposed to take:

- 0 or Null – No maximum time is specified
- Non-Null – The value specifies the maximum time allowed

If the maximum time expires, the method implementation may abort the execution. If it aborts, it shall return 2 (Error).

7.2.4.6.4 Return value

This method shall return one of the following return values:

Table 11 – RequestStateChange: Return values

Value	Description
0	The state change was successfully performed.
1	The method is not implemented.
2	An error has occurred.
4096	The request to change the state is being executed asynchronously, and the Job parameter references a ConcreteJob**Profile Error: A class adaptation "ConcreteJob" is referenced in a class adaptation link but is not defined or is defined more than once in this profile.** instance representing the request.

7.2.4.7 Operation: ModifyInstance( )

Requirement level: Conditional

Condition:

The SensorElementNameModification feature is implemented.

7.2.5 Adaptation: DiscreteSensor: CIM_Sensor

7.2.5.1 General

Adaptation type: Ordinary class

Implementation type: Instantiated

Requirement level: Mandatory

This adaptation models discrete sensors .

Table 12 – DiscreteSensor: Element requirements

Element	Requirement	Description
Base adaptations		
AbstractSensor	Optional	See AbstractSensor.
Properties		
CurrentState	Mandatory	See 7.2.5.2
PossibleStates	Mandatory	See 7.2.5.3

7.2.5.2 Property: CurrentState

370 **Requirement level:** Mandatory

371 **Constraint:**

372 OCL constraint in the context of a DiscreteSensor instance:

373
```
inv: self.PossibleStates->contains(self.CurrentState)
```

374

7.2.5.3 Property: PossibleStates

375 **Requirement level:** Mandatory

The set of array entry values of this array property shall represent the set of allowable values for the CurrentState property. The mapping between these values and the actual condition of the discrete sensor is implementation specific.

376 **Constraint:**

377 OCL constraint in the context of a DiscreteSensor instance:

378
```
inv:
self.SensorType = 2 /* Temperature */ implies
    self.PossibleStates->forAll( s |
        Set { "Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 3 /* Voltage */ implies
    self.PossibleStates->forAll( s |
        Set {"Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 4 /* Current */ implies
    self.PossibleStates->forAll( s |
        Set {"Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 5 /* Tachometer */ implies
    self.PossibleStates->forAll( s |
        Set {"Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 7 /* Switch */ implies
    self.PossibleStates->forAll( s |
        Set {"Closed", "Open", "Unknown" }->contains(s)) and
self.SensorType = 8 /* Lock */ implies
    self.PossibleStates->forAll( s |
        Set {"Locked", "Unlocked", "Unknown" }->contains(s)) and
self.SensorType = 9 /* Humidity */ implies
    self.PossibleStates->forAll( s |
        Set {"Humid", "Normal", "Unknown" }->contains(s)) and
self.SensorType = 10 /* Smoke Detection */ implies
    self.PossibleStates->forAll( s |
        Set {"Smokey", "Normal", "Unknown" }->contains(s)) and
self.SensorType = 11 /* Presence */ implies
    self.PossibleStates->forAll( s |
        Set {"Not Present", "Present", "Unknown" }->contains(s)) and
self.SensorType = 12 /* Air Flow */ implies
    self.PossibleStates->forAll( s |
        Set {"Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 13 /* Power Consumption */ implies
    self.PossibleStates->forAll( s |
        Set {"Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 14 /* Power Production */ implies
    self.PossibleStates->forAll( s |
        Set {"Bad", "Good", "Unknown" }->contains(s)) and
self.SensorType = 15 /* Pressure */ implies
```

```
self.PossibleStates->forAll( s |
    Set {"Bad", "Good", "Unknown" }->contains(s))
```

379 ### 7.2.6 Adaptation: NumericSensor: CIM_NumericSensor

380 #### 7.2.6.1 General

381 **Adaptation type:** Ordinary class

382 **Implementation type:** Instantiated

383 **Requirement level:** Mandatory

This adaptation models analog sensors .

384 **Table 13 – NumericSensor: Element requirements**

Element	Requirement	Description
Base adaptations		
AbstractSensor	Optional	See AbstractSensor.
Properties		
BaseUnits	Mandatory	
UnitModifier	Mandatory	
RateUnits	Mandatory	
CurrentReading	Mandatory	
LowerThresholdNonCritical	Conditional	See 7.2.6.2
UpperThresholdNonCritical	Conditional	See 7.2.6.3
LowerThresholdCritical	Conditional	See 7.2.6.4
UpperThresholdCritical	Conditional	See 7.2.6.5
LowerThresholdFatal	Conditional	See 7.2.6.6
UpperThresholdFatal	Conditional	See 7.2.6.7
SupportedThresholds	Mandatory	See 7.2.6.8
SettableThresholds	Mandatory	See 7.2.6.9
CurrentState	Mandatory	See 7.2.6.10
PossibleStates	Mandatory	See 7.2.6.11
Methods		
RestoreDefaultThresholds( )	Conditional	See 7.2.6.12
Operations		
ModifyInstance( )	Conditional	See 7.2.6.13

385 #### 7.2.6.2 Property: LowerThresholdNonCritical

386 **Requirement level:** Conditional

387 **Condition:**

The LowerThresholdNonCriticalSupported feature is implemented.

389 **Modification requirement:**

Conditional

390

Condition:

The LowerThresholdNonCriticalSettable feature is implemented.

7.2.6.3 Property: UpperThresholdNonCritical

Requirement level: Conditional

Condition:

The UpperThresholdNonCriticalSupported feature is implemented.

Modification requirement:

Conditional

Condition:

The UpperThresholdNonCriticalSettable feature is implemented.

7.2.6.4 Property: LowerThresholdCritical

Requirement level: Conditional

Condition:

The LowerThresholdCriticalSupported feature is implemented.

Modification requirement:

Conditional

Condition:

The LowerThresholdCriticalSettable feature is implemented.

7.2.6.5 Property: UpperThresholdCritical

Requirement level: Conditional

Condition:

The UpperThresholdCriticalSupported feature is implemented.

Modification requirement:

Conditional

Condition:

The UpperThresholdCriticalSettable feature is implemented.

7.2.6.6 Property: LowerThresholdFatal

Requirement level: Conditional

Condition:

The LowerThresholdFatalSupported feature is implemented.

Modification requirement:

Conditional

Condition:

The LowerThresholdFatalSettable feature is implemented.

7.2.6.7 Property: UpperThresholdFatal

Requirement level: Conditional

Condition:

The UpperThresholdFatalSupported feature is implemented.

Modification requirement:

Conditional

Condition:

The UpperThresholdFatalSettable feature is implemented.

7.2.6.8 Property: SupportedThresholds

Requirement level: Mandatory

The set of array entry values shall represent the set of thresholds that are implemented. If no thresholds are implemented, the array shall be empty.

7.2.6.9 Property: SettableThresholds

Requirement level: Mandatory

The set of array entry values shall represent the set of implemented thresholds that are modifiable.

If no thresholds are modifiable, the array shall be empty. The set of array entry values shall be a subset of the set of array entry values in the SupportedThresholds property.

7.2.6.10 Property: CurrentState

Requirement level: Mandatory

Constraint:

OCL constraint in the context of a NumericSensor instance:

```
inv: self.PossibleStates->contains(self.CurrentState)
```

7.2.6.11 Property: PossibleStates

Requirement level: Mandatory

The set of array entry values of this array property shall represent the set of allowable values for the CurrentState property. The mapping between these values and the actual condition of the analog sensor is implementation specific.

Constraint:

OCL constraint in the context of a NumericSensor instance:

```
define commonSet : Set =
    Set { "Non-Critical", "Lower Non-Critical", "Upper Non-Critical", "Critical",
        "Lower Critical", "Upper Critical", "Fatal", "Lower Fatal", "Upper
Fatal",
        "Normal", "Unknown" }
inv:
self.SensorType = 3 /* Voltage */ implies
```

```
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 4 /* Current */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 5 /* Tachometer */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 9 /* Humidity */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 10 /* Smoke Detection */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 11 /* Presence */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 12 /* Air Flow */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 13 /* Power Consumption */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 14 /* Power Production */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s)) and
self.SensorType = 15 /* Pressure */ implies
    self.PossibleStates->forAll( s | commonSet->contains(s))
```

7.2.6.12 Method: RestoreDefaultThresholds( )

Requirement level: Conditional

Condition:

At least one of the following is true:

- The LowerThresholdNonCriticalSettable feature is implemented.
- The UpperThresholdNonCriticalSettable feature is implemented.
- The LowerThresholdCriticalSettable feature is implemented.
- The UpperThresholdCriticalSettable feature is implemented.
- The LowerThresholdFatalSettable feature is implemented.
- The UpperThresholdFatalSettable feature is implemented.

Successful execution of this method shall reset the values of the thresholds of the sensor represented by an instance of this adaptation to its default values.

Table 14 – RestoreDefaultThresholds( ): Parameter requirements

Parameter	Description
Return value	See 7.2.6.12.1

7.2.6.12.1 Return value

This method shall return one of the following return values:

Table 15 – RestoreDefaultThresholds: Return values

Value	Description
0	The method has executed successfully.
1	The method is not implemented.
2	An error has occurred.

7.2.6.13 Operation: ModifyInstance( )

Requirement level: Conditional

Condition:

At least one of the following is true:

- The LowerThresholdNonCriticalSettable feature is implemented.
- The UpperThresholdNonCriticalSettable feature is implemented.
- The LowerThresholdCriticalSettable feature is implemented.
- The UpperThresholdCriticalSettable feature is implemented.
- The LowerThresholdFatalSettable feature is implemented.
- The UpperThresholdFatalSettable feature is implemented.

7.2.7 Adaptation: SensorCapabilities: CIM_EnabledLogicalElementCapabilities

7.2.7.1 General

Adaptation type: Ordinary class

Implementation type: Instantiated

Requirement level: Conditional

Condition:

At least one of the following is true:

- The SensorElementNameModification feature is implemented.
- The SensorStateManagement feature is implemented.

This adaptation models the capabilities of a sensor represented by the associated AbstractSensor instance, and of the implementation of that adaptation.

Table 16 – SensorCapabilities: Element requirements

Element	Requirement	Description
Properties		
InstanceID	Mandatory	Key
RequestedStatesSupported	Mandatory	See 7.2.7.2
ElementNameEditSupported	Mandatory	See 7.2.7.3
MaxElementNameLen	Conditional	See 7.2.7.4
Operations		
GetInstance( )	Mandatory	
EnumerateInstances( )	Mandatory	
EnumerateInstanceNames( )	Mandatory	
Associators( )	Mandatory	
AssociatorNames( )	Mandatory	
References( )	Optional	
ReferenceNames( )	Optional	

7.2.7.2 Property: RequestedStatesSupported

Requirement level: Mandatory

The set of array entry values of this property shall represent the set of supported requested states for the associated AbstractSensor instance.

Constraint:

OCL constraint in the context of a SensorCapabilities instance:

```
inv: if self.ElementCapabilities::ManagedElement.
    mrpIsFeatureAvailable('SensorStateManagement')
then (
    self.RequestedStatesSupported->forAll( entry |
        Set { 2 /* Enabled */, 3 /* Disabled */, 11 /* Reset */ }->
        contains(entry))
) else (
    self.RequestedStatesSupported->size() = 0
)
```

7.2.7.3 Property: ElementNameEditSupported

Requirement level: Mandatory

Constraint:

OCL constraint in the context of a SensorCapabilities instance:

```
inv: self.ElementCapabilities::ManagedElement.
    mrpIsFeatureAvailable('SensorElementNameModification') =
    self.ElementNameEditSupported
```

7.2.7.4 Property: MaxElementNameLen

Requirement level: Conditional

Condition:

The SensorElementNameModification feature is implemented.

7.2.8 Adaptation: ElementCapabilities: CIM_ElementCapabilities

7.2.8.1 General

Adaptation type: Association class

Implementation type: Instantiated

Requirement level: Conditional

Condition:

At least one of the following is true:

- The SensorElementNameModification feature is implemented.
- The SensorStateManagement feature is implemented.

This adaptation models the relationship between sensors that are represented by AbstractSensor instances and the capabilities of these sensors or their adaptation implementation that are represented by SensorCapabilities instances.

Table 17 – ElementCapabilities: Element requirements

Element	Requirement	Description
Properties		
ManagedElement	Mandatory	Key, see 7.2.8.2
Capabilities	Mandatory	Key, see 7.2.8.3
Operations		
GetInstance( )	Mandatory	

7.2.8.2 Property: ManagedElement

Requirement level: Mandatory

Reference kind: REF-typed

Constraints:

- Referenced instances shall be of class adaptation AbstractSensor.
- The multiplicity of this association end is 1 .. *

7.2.8.3 Property: Capabilities

Requirement level: Mandatory

Reference kind: REF-typed

Constraints:

- Referenced instances shall be of class adaptation SensorCapabilities.
- The multiplicity of this association end is 0 .. 1

7.2.9 Adaptation: SensoredElement: CIM_ManagedSystemElement

This adaptation models sensored elements that are measured by sensors represented by associated AbstractSensor instances. AbstractSensor instances representing sensors that sensor the entire computer system hosting them (rather than elements within that computer system) shall not be associated that way.

Adaptation type: Ordinary class

Implementation type: Abstract

Requirement level: Defined by its derived adaptations

Table 18 – SensoredElement: Element requirements

Element	Requirement	Description
Operations		
Associators( )	Mandatory	
AssociatorNames( )	Mandatory	
References( )	Mandatory	
ReferenceNames( )	Mandatory	

7.2.10 Adaptation: AssociatedSensor: CIM_AssociatedSensor

7.2.10.1 General

Adaptation type: Association class

Implementation type: Instantiated

Requirement level: Mandatory

This adaptation models the relationship between sensors that are represented by AbstractSensor instances and the sensored elements measured by these sensors that are represented by SensoredElement instances.

Table 19 – AssociatedSensor: Element requirements

Element	Requirement	Description
Properties		
Antecedent	Mandatory	Key, see 7.2.10.2
Dependent	Mandatory	Key, see 7.2.10.3
Operations		
GetInstance( )	Mandatory	

7.2.10.2 Property: Antecedent

Requirement level: Mandatory

Reference kind: REF-typed

Constraints:

- Referenced instances shall be of class adaptation AbstractSensor.
- The multiplicity of this association end is 1 .. *

7.2.10.3 Property: Dependent

Requirement level: Mandatory

Reference kind: REF-typed

Constraints:

- Referenced instances shall be of class adaptation SensoredElement.
- The multiplicity of this association end is 0 .. *

8 Use cases and state descriptions

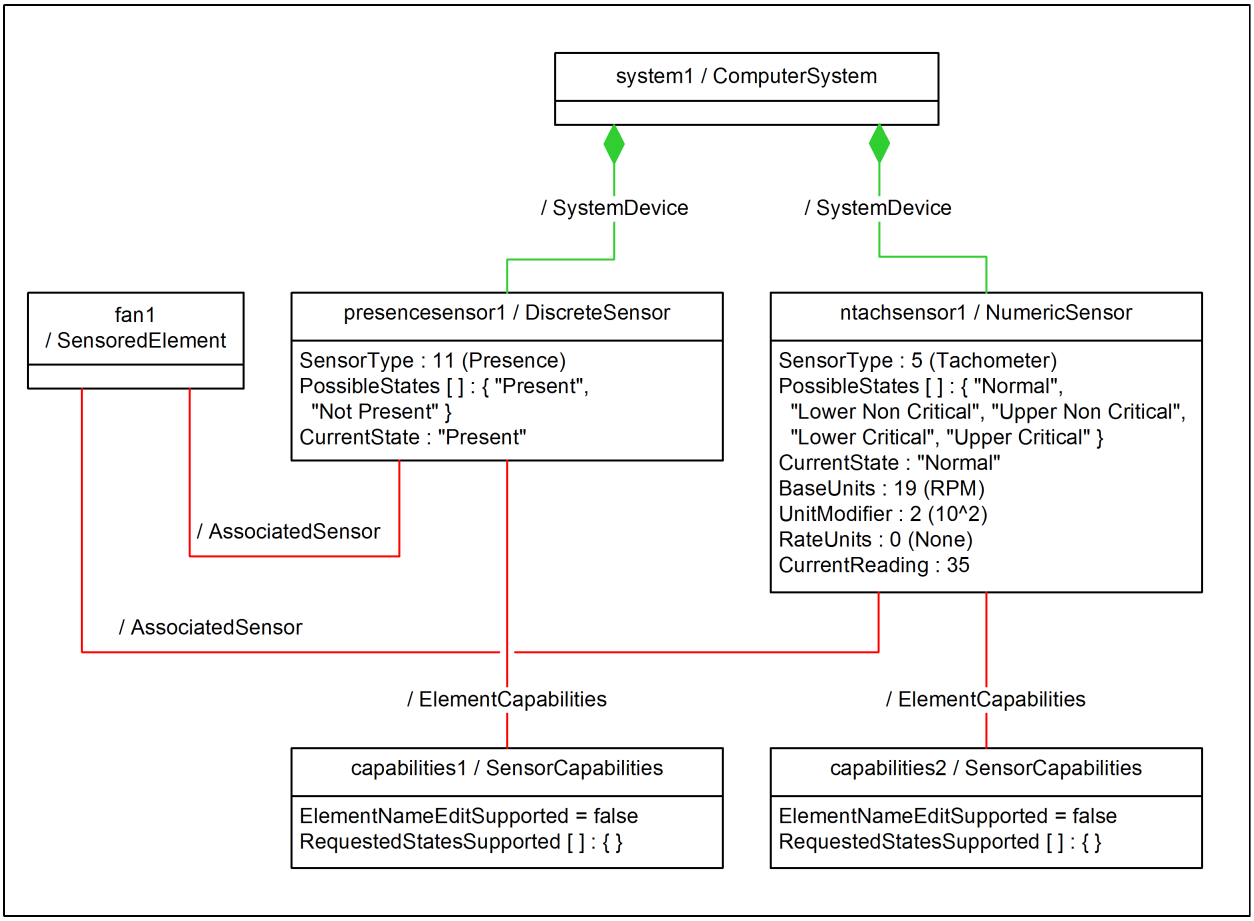
8.1 State description: SimpleObjectDiagram

The following figure shows a simple object diagram that represents a scenario that conforms to the requirements of this profile. In that scenario, there is one sensored element that is a fan, represented by an instance of CIM_Fan in the role of the SensoredElement adaptation. That fan has two sensors, a discrete sensor represented by the DiscreteSensor instance presencesensor1, and an analog sensor represented by the NumericSensor instance ntachsensor1. Both instances are associated with the SensoredElement instance through AssociatedSensor instances.

Based on the value 11 (Presence) of the SensorType property of the DiscreteSensor instance presencesensor1, the sensor represented by that instance is a presence sensor for the fan. In this implementation, only the values “Present” and “Not Present” are supported for the PossibleStates property.

540 Based on the value 5 (Tachometer) of the SensorType property of the NumericSensor instance ntachsensor1, the sensor represented by that instance is a speed sensor for the fan that provides numeric reading of the fan speed. Based on the value of the BaseUnits property (value 19 = RPM), and the RateUnits property showing no additional units, the fan speed is represented in RPM units. The CurrentReading property in this scenario has a value of 35, which is multiplied by the unit modifier represented by the UnitModifier property (value 2, so the actual multiplier is 10^2), to calculate the fan speed of 3500 RPM.

541



542 **Figure 3 – Simple object diagram**

543

8.2 Use case: ShowAllSensorStates

544 This use case describes the flow to obtain the current state of all sensored elements in a computer system, possibly including the computer system itself.

545 This use case has the following preconditions:

546 • A ComputerSystem instance is known.

547 The main flow for this use case consists of the following steps:

548 1. Invoke the Associators( ) for SystemDevice operation on that ComputerSystem instance. This will retrieve all DiscreteSensor and NumericSensor instances representing sensors that are hosted by the computer system represented by that ComputerSystem instance.

549 2. Iterate through these retrieved instances and inspect their DiscreteSensor . CurrentState and NumericSensor . CurrentState property values, which will represent the states of their sensored elements .

550

8.3 Use case: FindSensorsOfSystemElement

This use case describes the flow to find the sensors for a given sensored element in a computer system.

551 This use case has the following preconditions:

552 • A SensoredElement instance is known.

553 The main flow for this use case consists of the following step:

554 1. Invoke the Associators( ) for AssociatedSensor operation on that SensoredElement instance. This will retrieve all DiscreteSensor and NumericSensor instances representing sensors that measure the sensored element represented by that SensoredElement instance.

555

8.4 Use case: ChangeThreshold

This use case describes the flow to change the upper non-critical threshold of a given analog sensor .

556 This use case has the following preconditions:

557 • A NumericSensor instance is known.

558 The main flow for this use case consists of the following steps:

559 1. Invoke the GetInstance( ) operation on that NumericSensor instance. This step may be skipped if the instance has been retrieved by other means already.

560 2. Determine whether the SettableThresholds property in that instance contains an array entry value of 1 (UpperThresholdNonCritical).

562 3. If so, the threshold may be changed. Change the threshold by invoking the ModifyInstance( ) operation on that NumericSensor instance, including the new value for the UpperThresholdNonCritical property in the modified instance.

563 Otherwise, the threshold cannot be changed.

564

8.5 Use case: ResetThresholds

This use case describes the flow to reset the thresholds of a given analog sensor .

565 This use case has the following preconditions:

566 • A NumericSensor instance is known.

567 The main flow for this use case consists of the following step:

568 1. Invoke the RestoreDefaultThresholds( ) method on that NumericSensor instance.

569

8.6 Use case: DetermineElementNameModifiability

This use case describes the flow to determine whether the ElementName property of a given sensor can be modified.

570 This use case has the following preconditions:

- A DiscreteSensor or NumericSensor instance is known.

The main flow for this use case consists of the following steps:

1. Retrieve the associated SensorCapabilities instance as follows:
    - Invoke the Associators( ) for ElementCapabilities operation on that DiscreteSensor or NumericSensor instance.
2. Inspect the value of the ElementNameEditSupported property of that SensorCapabilities instance.

   If that value is TRUE, the client can modify the ElementName property of the given DiscreteSensor or NumericSensor instance.

8.7 Use case: DetermineStateManagementSupport

This use case describes the flow to determine whether state management (i.e. the SensorStateManagement feature) is supported for a given sensor .

This use case has the following preconditions:

- A DiscreteSensor or NumericSensor instance is known.

The main flow for this use case consists of the following steps:

1. Retrieve the associated SensorCapabilities instance as follows:
    - Invoke the Associators( ) for ElementCapabilities operation on that DiscreteSensor or NumericSensor instance.
2. Inspect the value of the RequestedStatesSupported array property of that SensorCapabilities instance.

   If the array property contains at least one array entry, state management is supported for the sensor represented by the given DiscreteSensor or NumericSensor instance.

ANNEX A
(informative)

Change log

588

Version	Date	Description
1.0.0c	2006-05-16	DSP1009: Released as Preliminary Standard
1.0.0	2007-11-06	DSP1009: Released as Final Standard
1.0.1	2008-09-25	DSP1009: Released as DMTF Standard
1.0.2	2009-10-28	DSP1009: Released as DMTF Standard, with the following changes: • Changed the values for the EnabledState property
1.0.3m	2013-08-01	XMP1009: Included as a sample profile into DSP2023

589

590

Bibliography

591 The following documents may provide additional background.

592 DMTF DSP1000, *Management Profile Specification Template 1.1*, http://www.dmtf.org/standards/published_documents/DSP1000_1.1.pdf