Document Number: DSP1033

Date: 2007-07-31

Version: 1.0.0

Profile Registration

Document Type: Specification

Document Status: Final

Document Language: E

CONTENTS

Foreword 5
Introduction 6
1 Scope 7
2 Normative References 7
  2.1 Approved References 7
  2.2 References Under Development 7
  2.3 Other References 7
3 Terms and Definitions 7
4 Symbols and Abbreviated Terms 10
5 Synopsis 10
6 Description (Informative) 11
  6.1 Central and Scoping Class Overview 11
  6.2 Central Class Profile Implementation Advertisement 12
  6.3 Scoping Class Profile Implementation Advertisement 13
7 Implementation 15
  7.1 Relationship between Interop and Implementation Namespaces 15
  7.2 Establishing a Consistent Interop Namespace 15
  7.3 Cross-Namespace Associations 15
  7.4 Implementing Central Class or Scoping Class Methodologies 15
  7.5 Central Class and Central Instance Identification 16
  7.6 Scoping Class and Scoping Instance Identification 16
  7.7 Association Traversal Path Existence 16
  7.8 Overlapping Profile Definitions 17
  7.9 CIM_RegisteredProfile 18
  7.10 CIM_ElementConformsToProfile 18
  7.11 CIM_ReferencedProfile 18
8 Methods 19
  8.1 Profile Conventions for Operations 19
  8.2 CIM_RegisteredProfile 19
  8.3 CIM_ReferencedProfile 20
  8.4 CIM_ElementConformsToProfile 20
9 Use Cases (Informative) 20
  9.1 Object Diagrams 20
  9.2 Retrieve the Profile Information for an Instance of CIM_ComputerSystem 22
  9.3 Retrieve the Profile Version Information for a Specific Fan 22
  9.4 General Algorithm for Retrieving Profile Information 22
  9.5 Using an Association Path Traversal to Determine Conformance 24
  9.6 Enumerate Profiles Advertised in Interop Namespace by an Implementation 26
  9.7 Determine Top-Level Profiles in an Interop Namespace 26
  9.8 Determining Implementation Instances for a Profile 27
  9.9 Peer Component Profile Relationships 29
  9.10 Determining Whether Central or Scoping Class Methodology Is in Use 30
  9.11 Example of Profile Compliance Hierarchy 30
10 CIM Elements 32
  10.1 CIM_RegisteredProfile 32
  10.2 CIM_ElementConformsToProfile 32
  10.3 CIM_ReferencedProfile 33
ANNEX A (informative) Change Log 34
ANNEX B (informative) Acknowledgements 35

Figures

Figure 1 – Profile Registration Profile: Class Diagram 11
Figure 2 – Central Class Implementation Conformance Traversal Example 13
Figure 3 – Scoping Class Implementation Conformance Traversal Example 14
Figure 4 – Profile Registration Profile: Object Diagram 21
Figure 5 – Redundant Fan: Object Diagram 25
Figure 6 – Peer Component Profiles: Object Diagram 29
Figure 7 – Profile Compliance Hierarchy 31

Tables

Table 1 – Operations: CIM_RegisteredProfile 19
Table 2 – Operations: CIM_ReferencedProfile 20
Table 3 – Operations: CIM_ElementConformsToProfile 20
Table 4 – CIM Elements: Profile Registration Profile 32
Table 5 – Class: CIM_RegisteredProfile 32
Table 6 – Class: CIM_ElementConformsToProfile 33
Table 7 – Class: CIM_ReferencedProfile 33

Foreword

The *Profile Registration* (DSP1033) was prepared by the DMTF WBEM Infrastructure & Protocols - Profiles Working Group.

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

Introduction

The *Profile Registration* defines the classes used to describe the DMTF profile registration and the version information of the profiles advertised as implemented for a managed system and components of the system. The information in this specification is intended to be sufficient for a provider or consumer of this data to identify unambiguously the classes, properties, methods, and values that must be instantiated to represent the profile name, version, and owning organization information that is modeled using the DMTF Common Information Model (CIM) Schema.

Profile specifications that normatively describe the behavior and use of DMTF CIM classes for the representation of specific management domains are being defined and published by the DMTF, Storage Network Industry Association (SNIA), and other organizations. Mechanisms for CIM data-model-based system and component management instrumentation are needed to advertise what profiles the instrumentation has implemented. This document covers the representation of the profiles and profile versions implemented and how specific CIM class instances can be identified as having been implemented per normative definitions in a specific profile.

The target audience for this specification is implementers who are developing products that publicize management information using CIM or consumers of CIM-based management information.

Profile Registration

1 Scope

The *Profile Registration* extends the management capability of the referencing profiles by adding the capability to describe the registration and versioning of Common Information Model (CIM) profiles that are implemented by CIM-based system and component-management instrumentation.

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 2.3.0*

DMTF DSP0200, *CIM Operations over HTTP 1.2.0*

DMTF DSP1000, *Management Profile Specification Template*

DMTF DSP1001, *Management Profile Specification Usage Guide*

2.2 References Under Development

DMTF DSP1004, *Base Server Profile*, version 1.0 Preliminary

DMTF DSP1009, *Sensor Profile*, version 1.0 Preliminary

DMTF DSP1013, *Fan Profile*, version 1.0 Preliminary

DMTF DSP1015, *Power Supply Profile*, version 1.0 Preliminary

2.3 Other References

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*

OMG, *Unified Modeling Language (UML) from the Open Management Group (OMG)*

OMG, UML Specifications

3 Terms and Definitions

For the purposes of this document, the following terms and definitions apply. For the purposes of this document, the terms and definitions given in DSP1001 also 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

autonomous profile

a profile that defines an autonomous and self-contained management domain

An autonomous profile may be used alone. An autonomous profile may optionally reference other profiles, including component profiles and other autonomous profiles.

3.15

Central Class

a class defined in a profile and identified as the focal point for identifying conformance with that profile

3.16

Central Instance

an instance of the Central Class that is the focal point for an implementation of the profile

3.17

component profile

a profile that describes a subset of a management domain

The profile specification of a component profile includes CIM elements that are scoped within an autonomous profile (or in some cases, another component profile).

3.18

Interop Namespace

a namespace in which the CIM_RegisteredProfile instances are instantiated in order to advertise the availability of a conformant implementation of a profile

3.19

implementation Namespace

a namespace in which the classes and instances that compose the advertised profile conformance are implemented

3.20

Scoping Class

a class defined in a referencing profile and identified as the top-level class in an implementation hierarchy that is associated with the representation of the referencing profile and is the algorithmic focal point for identifying profile conformance when using the Scoping Class Methodology

3.21

Scoping Instance

an instance of the Scoping Class

3.22

subject profile

the implemented profile for which the instances of the classes defined in the *Profile Registration* are being used to advertise profile implementation

3.23

referencing profile

a profile that includes a reference to another profile in its Related Profiles section.

3.24
referenced profile
a profile that is included in another profile's Related Profiles section.

4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

4.1
CIM
Common Information Model

4.2
ECTP
CIM_ElementConformsToProfile

4.3
NIC
network interface card

4.4
OM
Object Manager

4.5
SMIRL
Storage Management Initiative Recipe Language

4.6
UML
Unified Modeling Language

4.7
WBEM
Web-Based Enterprise Management

5 Synopsis

Profile Name: *Profile Registration*

Version: 1.0.0

Organization: DMTF WBEM Infrastructure & Protocols - Profiles Working Group

CIM schema version: 2.10.0

Central Class: CIM_RegisteredProfile

Scoping Class: CIM_RegisteredProfile

6 Description (Informative)

The *Profile Registration* describes the necessary properties and methods to represent profile and profile versioning implementation conformance.

Figure 1 represents the Unified Modeling Language (UML) class diagram for the *Profile Registration*. See 7.1 for more normative specifications for namespaces.

For simplicity, the prefix *CIM_* has been removed from the names of the classes.

A profile is represented by an instance of the CIM_RegisteredProfile class as shown in Figure 1. That instance specifies the profile name, owning organization, and the profile version with which the implementation is compliant.

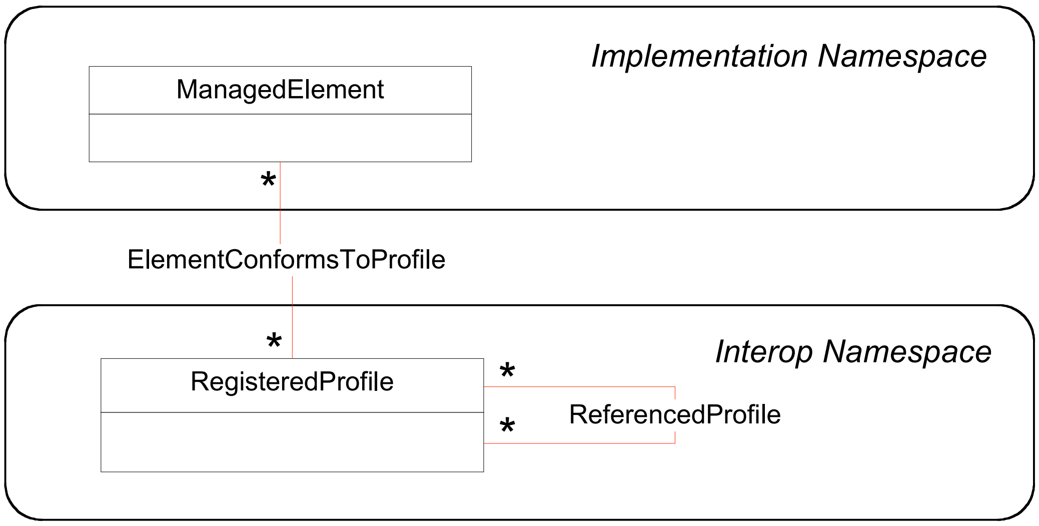


Figure 1 – Profile Registration: Class Diagram

6.1 Central and Scoping Class Overview

The *Profile Registration* establishes the concept of a Central Class and Central Instance. Profiles typically include constraints and behavioral requirements for more than one CIM element. For an implementation to advertise conformance with a profile, each of the implemented, related elements defined in the profile must be conformant with the profile specification.

Another concept is the Scoping Class and Scoping Instance. The Scoping Class is a class typically identified as the Central Class in a referencing profile. In autonomous profiles, the Central Class and the Scoping Class are the same. The Scoping Instance is identified as the top-level class instance in an implementation hierarchy that is associated with the instance of CIM_RegisteredProfile that represents the referencing profile and is an algorithmic focal point for identifying advertised profile implementation conformance.

Thus, if a client with *a priori* knowledge of the elements defined in the profile determines that one element is conformant with the profile, this is sufficient for the client to know that the related elements are also conformant. The *Profile Registration* takes advantage of this condition to specify the client algorithms or methodologies that are used to determine with which profile an implemented element is conformant.

Instances of the CIM_RegisteredProfile class are used to identify the profile specification that instrumentation is advertising as being implemented. This information includes the profile name, owning organization, and the profile version with which the implementation is compliant.

Instances of CIM_ElementConformsToProfile are used to associate instances of Central and Scoping Classes defined in profiles with the CIM_RegisteredProfile that identifies the particular profile specifications that are implemented.

The *Profile Registration* defines two methodologies through which a provider can advertise implementation conformance with a particular profile.

- The first methodology is hereafter referred to as the *Central Class methodology* and is characterized by a CIM_ElementConformsToProfile association between every instance of a profile's Central Class and the instance of CIM_RegisteredProfile that represents the profile. See 6.2 and 7.4 for more information about the Central Class methodology.
- The second methodology is hereafter referred to as the *Scoping Class methodology* and uses the CIM_ElementConformsToProfile association only between the "top-level" or Scoping Class instance of a connected set of instances and the instance of CIM_RegisteredProfile that represents the "top-level" or autonomous profile in a profile compliance hierarchy. In the Scoping Class methodology, the Central Class instances of the component profiles are not associated through CIM_ElementConformsToProfile to instances of CIM_RegisteredProfile that represent the component profiles. See 6.3 and 7.4 for more information about the Scoping Class methodology.

The CIM_ManagedElement shown in Figure 1 represents either the Central Class or the Scoping Class of a profile.

The Central Class and Scoping Class methodologies for advertising profile implementation conformance are mutually exclusive for a specific profile (and profile version) being advertised. However, an implementation situation in which several versions of the same profile are implemented may use both methods simultaneously if the Scoping Class methodology is used for one of the profile versions implemented. The use of both methodologies is recommended for the following situation:

> Two (or more) versions of the same profile have been implemented, two (or more) instances of CIM_RegisteredProfile have been instantiated representing the two (or more) different versions of the same profile, and the Scoping Class methodology was used to advertise one profile version implementation. In this case, one profile version would have been advertised through the Scoping Class methodology, while the others would be advertised through the Central Class methodology.

This approach addresses the problem of determining, through the Scoping Class methodology, which profile a specific managed element implementation is conformant with when multiple versions of a profile are implemented. An example of this situation could be a system with two NICs, each from a different vendor that has delivered a provider for the *Ethernet Port Profile* where the implementations are of different versions of the profile.

6.2 Central Class Profile Implementation Advertisement

The Central Class profile implementation advertisement methodology is based on a straightforward approach whereby every instance of the specified Central Class of an implemented profile is associated through the CIM_ElementConformsToProfile association to an instance of CIM_RegisteredProfile that represents the specific profile and version with which the implementation is advertising conformance.

This method is straightforward because client applications only need to traverse the CIM_ElementConformsToProfile association from or to the profile's Central Class instance to ascertain the profiles to which the implementation advertises conformance. The ability to traverse associations across namespaces (not represented in Figure 2) is subject to implementation requirements defined in 7.3.

Figure 2 provides an example of the Central Class methodology of advertising profile implementation conformance. In the figure, the dotted line bi-directional arrows represent the ability of an application to traverse the CIM_ElementConformsToProfile association in the following ways:

- from the instance of the Central Class identified in the profile to the instance of CIM_RegisteredProfile that represents the profile
- from an instance of CIM_RegisteredProfile that represents the implemented profile to the instances of the Central Class identified in the profile

For simplicity, the prefix *CIM_* has been removed from the names of the classes.

In Figure 2, the CIM_ComputerSystem, CIM_Fan, and CIM_Sensor classes are the Central Classes for the profiles represented by instances of the CIM_RegisteredProfile class.

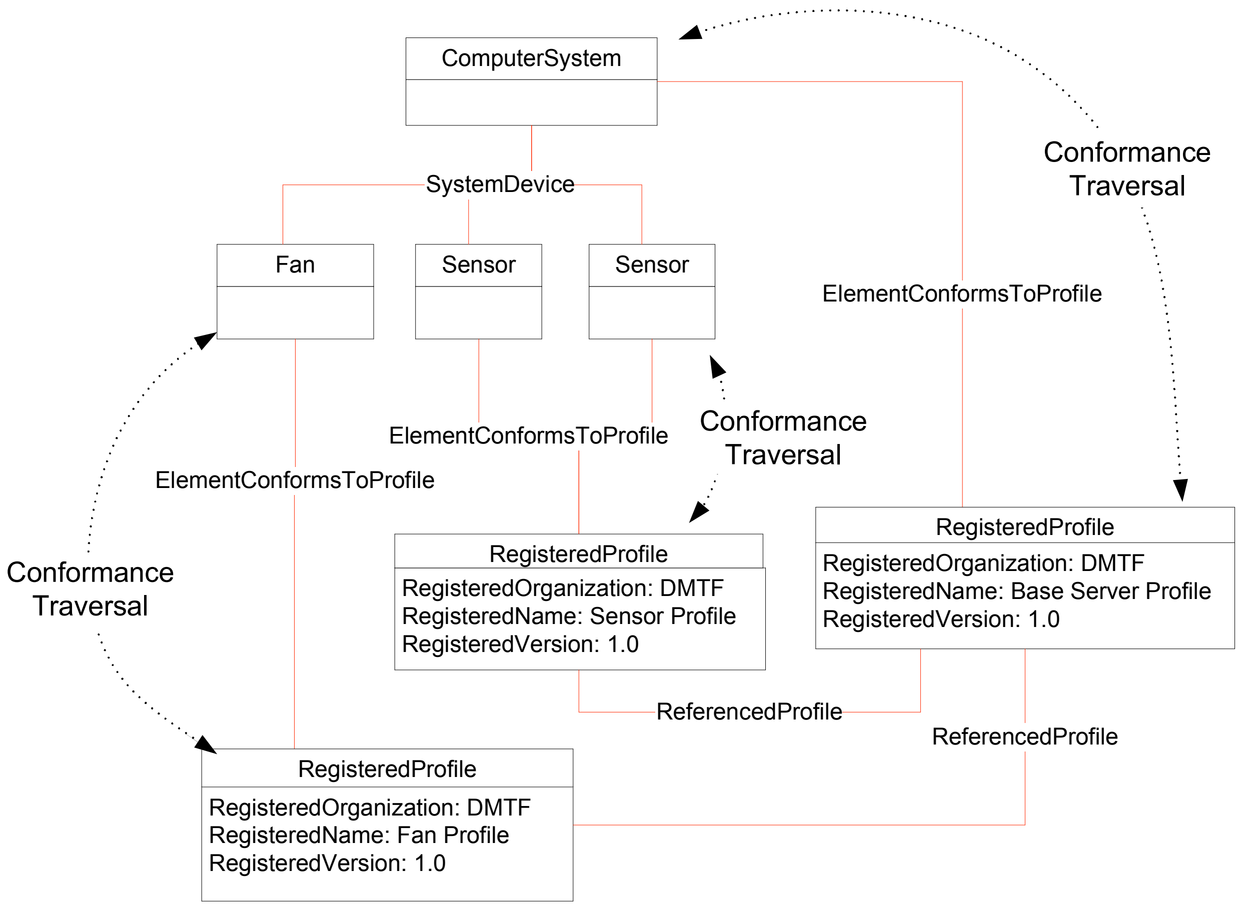


Figure 2 – Central Class Implementation Conformance Traversal Example

6.3 Scoping Class Profile Implementation Advertisement

The Scoping Class profile implementation advertisement methodology is an approach characterized by the use of the CIM_ElementConformsToProfile association only between the “top-level” or Scoping Class instance in an implementation class hierarchy and the “top-level” or autonomous profile representation in a profile implementation conformance class hierarchy.

Figure 3 provides an example of the Scoping Class methodology of advertising profile implementation conformance.

For simplicity, the prefix *CIM_* has been removed from the names of the classes.

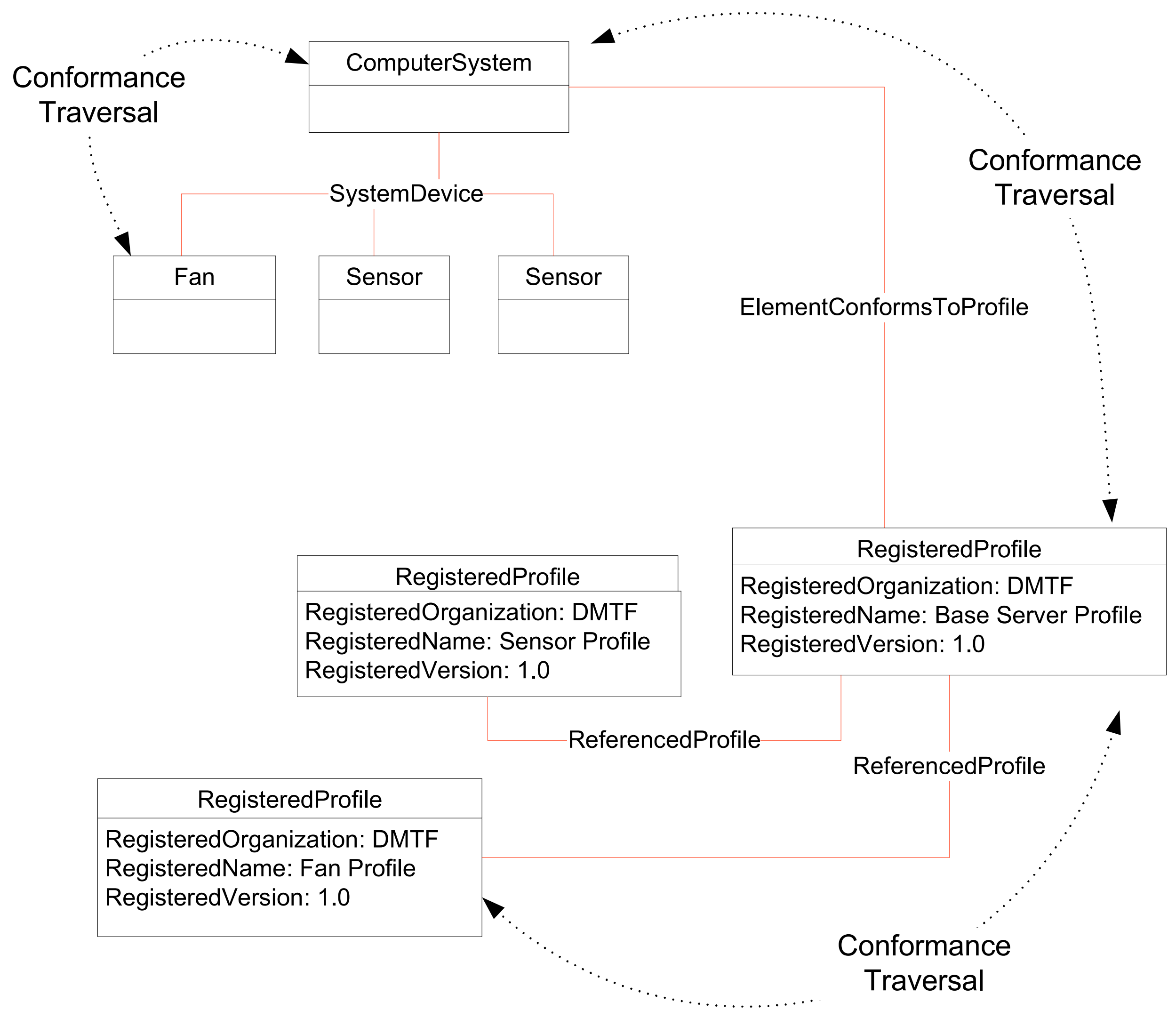


Figure 3 – Scoping Class Implementation Conformance Traversal Example

In Figure 3, the client application may traverse from an instance of CIM_Fan to the DMTF *Fan Profile* Scoping Instance, CIM_ComputerSystem, through the CIM_SystemDevice association. Then the CIM_ElementConformsToProfile association is traversed to an instance of CIM_RegisteredProfile that represents the *Base System Profile*. Finally, the CIM_ReferencedProfile association is traversed to an instance of CIM_RegisteredProfile that represents the DMTF *Fan Profile* with which the implementation of the CIM_Fan instance is advertising conformance.

The client application may reverse this traversal and start from the instance of CIM_RegisteredProfile that represents the DMTF *Fan Profile* to get to the instance of CIM_Fan. The ability to traverse associations across namespaces (not represented in the figure) is subject to implementation requirements defined in 7.3.

7 Implementation

This section details the requirements related to the arrangement of instances and their properties for implementations of this profile.

7.1 Relationship between Interop and Implementation Namespaces

An Interop Namespace and an Implementation Namespace may be the same.

NOTE: a simple configuration may have a single Interop Namespace and a single Implementation Namespace that are the same. A more complex configuration may have a single Interop Namespace and multiple Implementation Namespaces that are all not the same. A profile implementation may span multiple namespaces.

A typical configuration will have a single Interop Namespace and one or more Implementation Namespaces. An Interop Namespace should be established separate from Implementation Namespaces.

A profile implementation may span multiple Implementation Namespaces.

See clause 3, "Terms and Definitions", for definitions of the terms *Interop Namespace* and *Implementation Namespace*. Defining how namespaces are established is beyond the scope of this profile.

7.2 Establishing a Consistent Interop Namespace

An implementation of this profile shall include an Interop Namespace. The name of this Interop Namespace shall be either “interop” (preferred) or “root/interop”. A slash character (/) may precede the name of the Interop Namespace. The name of the Interop Namespace preceded with a slash, when used as an identification of a namespace either as a parameter to a CIM operation or as a CIM Instance property value, shall be considered by implementations as equivalent to the Interop Namespace without the preceding slash.

The purpose of the Interop Namespace is to provide a common and well-known place for a client application to discover all of the profiles that are supported within a given CIM Server.

The existence of a conformant implementation of a particular profile shall be advertised through instances of the CIM_RegisteredProfile class together with CIM_ReferencedProfile associations in the Interop Namespace. Instances of the CIM_ElementConformsToProfile association shall be used to associate instances of CIM_RegisteredProfile with instances of Central Classes defined in subject profiles according to the Central Class or Scoping Class methodology.

7.3 Cross-Namespace Associations

Associations that cross namespaces within the same CIMOM shall be instantiated in both namespaces. The rationale for this is to support association traversal from either namespace back to the other.

NOTE: Each of these association instances has their class exist in the same namespace as the association instance. The versions of these two association classes may be different.

7.4 Implementing Central Class or Scoping Class Methodologies

Implementations shall use either one or both Central Class and Scoping Class methodologies for advertising implementation conformance.

In situations in which implementations have small footprint requirements and want to reduce the number of instances or in which the implementation is monolithic and the version of specific profiles is homogeneous, the implementation may use the Scoping Class methodology and reduce the number of necessary CIM_ElementConformsToProfile associations.

In situations in which multiple versions of the same profile may be implemented, such as multi-vendor providers being integrated into a single CIM Object Manager (OM) for server management, the Central Class methodology is recommended to provide unambiguous relationships through CIM_ElementConformsToProfile between Central Class instances and the instance of CIM_RegisteredProfile that advertises the version and profile implemented.

Implementations do not advertise which methodology is used because the methodology can be ascertained by testing whether the Central Class instance has a CIM_ElementConformsToProfile association (see 9.6).

7.5 Central Class and Central Instance Identification

A subject profile that uses the Central Class methodology shall identify at least one Central Class and should identify exactly one Central Class.

The subject profile shall be written in such a way as to ensure that an implementation of the profile has exactly one Central Instance. An implementation of the profile is the instantiation of a set of connected CIM object instances that represent the system or component being instrumented. It is expected that multiple implementations of the profile will be instrumented to represent multiple similar systems or components. The Central Instances of these multiple implementations should be associated to a single instance of CIM_RegisteredProfile that represents the subject profile.

An autonomous subject profile shall define the Scoping Instance and the Central Instance as the same instance.

A conformant implementation that uses the Central Class methodology to advertise profile conformance shall ensure that a CIM_ElementConformsToProfile association is instantiated between the Central Instance of CIM_ManagedElement and the profile instance of CIM_RegisteredProfile.

7.6 Scoping Class and Scoping Instance Identification

A subject profile shall identify exactly one Scoping Class.

The subject profile shall ensure that all conformant instances of the Central Class defined by the profile shall be connected to the appropriate instances of the Scoping Class through a connected set of classes and associations according to well-defined association traversal algorithms. Additionally, a conformant implementation shall ensure that no non-conformant Central Instance is connected to the Scoping Instance through other association traversal algorithms of the subject profile.

An autonomous subject profile shall identify the Scoping Class and the Central Class to be the same and it shall identify only one Central Class instance.

A conformant implementation that uses the Scoping Class methodology shall ensure that a CIM_ElementConformsToProfile association is instantiated between the Scoping Instance of CIM_ManagedElement and the instance of CIM_RegisteredProfile that represents the profile that identifies the Scoping Instance as the Central Class.

7.7 Association Traversal Path Existence

The subject profile shall identify an association path from instances of the Central Class to all other implemented elements. When the association path is non-trivial or unobvious, the subject profile shall explicitly enumerate the association traversal algorithms that form the association paths.

- For non-Central Instances specified by a subject profile that are directly associated with the Central Instance of that profile, if the associations are defined by the subject profile, and there are no other constraints, an additional algorithm shall not be required because the simple association is sufficient.

- For all other instances of classes specified by a subject profile that are not Central Instances or directly associated with the Central Instance, a traversal algorithm shall be defined by the subject profile to enable a client to traverse by means of associations and classes between the particular class instances and the Central Instance.

The traversal algorithm between the Scoping Class instance and the Central Class instance shall be explicitly defined by the subject profile when it is non-trivial or unobvious. When starting with the Scoping Class instance, the subject profile shall specify such path traversal definitions or other algorithms that are unobvious.

7.8 Overlapping Profile Definitions

The three cases of overlapping profile definitions are as follows:

1) Multiple profiles define the same class.
2) Multiple versions of a profile are implemented for the same scoping system.
3) A profile further constrains the definition of a class defined in referenced profiles.

7.8.1 Multiple Profiles Define the Same Class

Subject profiles should be orthogonally defined such that the Central Class in the profile is unique to the subject profile. Generally, the exception is the definition of CIM_System and subclasses that are typically used as Central Classes in autonomous profiles and have CIM_ElementConformsToProfile associations to the instance of CIM_RegisteredProfile that represents the autonomous profile.

Classes other than the Central Class of a profile may be defined in other profiles. Examples of these non-Central Classes are CIM_EnabledLogicalElementCapabilities and CIM_RedundancySet. Instances of these non-Central Classes can be identified as belonging to a particular profile implementation through traversal of association paths defined in the subject profile (see 7.7). An example of this situation is the traversal of the CIM_MemberOfCollection associations from an instance of CIM_RedundancySet where instances of CIM_Fan are found as members of the CIM_RedundancySet collection. In this case, the instance of CIM_RedundancySet in question shall behave as normatively defined in the DMTF *Fan Profile*.

7.8.2 Multiple Versions of the Same Profile Are Implemented

When multiple overlapping implementations of a subject profile exist, the subject profile shall constrain the implementation such that all CIM elements identified in the subject profile in a particular implementation are conformant with the same version of the subject profile. Note that the preference is for subject profiles to be defined in such a way that the Central Class of the subject profile is the point of intersection for the potentially overlapping implementations. This would allow implementations of different versions to be indicated with specific ECTP associations per the Central Class methodology.

7.8.3 Profile Constrains the Class Defined in Referenced Profiles

Instances of CIM_ManagedElement defined in the subject profile or in profiles referenced by the subject profile shall constitute a set of connected instances from the Central Instance. If the set of connected instances overlaps (that is, profile A defines something about a class in profile B, and profile A references profile B), the implementation of profile B shall conform to definitions in profile A. Each profile defines constraints on a set of classes and instances of these classes. If an implementation implements the same class from multiple profiles, each conformant instance shall adhere to the rules set forth in all the related profiles.

7.9 CIM_RegisteredProfile

This clause defines required properties of CIM_RegisteredProfile.

7.9.1 CIM_RegisteredProfile.RegisteredOrganization

The CIM_RegisteredProfile.RegisteredOrganization property shall represent the organization that owns the profile specification to which the implementation adheres.

For DMTF profiles, the value shall be 2 (“DMTF”).

7.9.2 CIM_RegisteredProfile.RegisteredName

The CIM_RegisteredProfile.RegisteredName property shall represent the name of the profile that has been assigned by the organization represented in the CIM_RegisteredProfile.RegisteredOrganization property. The label “Profile” shall not be included in the RegisteredName property as the last part of the name of the profile.

EXAMPLE: The name “Base Server” shall be used instead of “Base Server Profile”.

RegisteredName matches (pattern “.+”).

7.9.3 CIM_RegisteredProfile.RegisteredVersion

The CIM_RegisteredProfile.RegisteredVersion property shall represent the version of the profile that has been identified in the CIM_RegisteredProfile.RegisteredName property.

RegisteredVersion matches (pattern “`^(([123456789][0123456789]*)|0)\.(([123456789][0123456789]*)|0)\.(([123456789][0123456789]*)|0`")

7.10 CIM_ElementConformsToProfile

An instance that represents a particular CIM_ElementConformsToProfile association shall be instantiated in both the Interop Namespace and the Implementation Namespace when these namespaces are not the same. See 7.2 for more information about establishing a consistent Interop Namespace.

In the case where the Interop Namespace and the Implementation Namespace are the same, only one instance of a particular CIM_ElementConformsToProfile association shall be instantiated.

7.11 CIM_ReferencedProfile

A profile specification may include references to other profile specifications in order to define relationships between the elements defined in the various related profiles. The CIM_ReferencedProfile association shall be used to represent the relationship between profiles. Note that the use of the Dependent and Antecedent properties in the CIM_ReferencedProfile association is defined in a somewhat non-intuitive way, such that the profile being referenced is the antecedent and the profile doing the referencing is the dependent.

7.11.1 CIM_ReferencedProfile.Dependent

The CIM_ReferencedProfile.Dependent property of the CIM_ReferencedProfile association instance shall be set to the value of a reference to the CIM_RegisteredProfile instance for the referencing profile.

7.11.2 CIM_ReferencedProfile.Antecedent

The CIM_ReferencedProfile.Antecedent property of the CIM_ReferencedProfile association instance shall be set to the value of a reference to the CIM_RegisteredProfile instance for the referenced profile.

8 Methods

This section details the requirements for supporting intrinsic operations for the CIM elements defined by this profile. No extrinsic methods are defined by this profile.

8.1 Profile Conventions for Operations

Support for operations for each profile class (including associations) is specified in the following subclauses. Each of these subclauses includes either a statement "All operations are supported as described by DSP0200 version 1.2" or a table listing all the operations that are not supported by this profile or where the profile requires behavior other that described by DSP0200 version 1.2.

The default list of operations is as follows:

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

A compliant implementation shall support all the operations in the default list for each class, unless the "Requirement" column states something other than *Mandatory*.

8.2 CIM_RegisteredProfile

Table 1 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 1 – Operations: CIM_RegisteredProfile

Operation	Requirement	Messages
GetInstance	Mandatory	None
ModifyInstance	Unspecified	None
Associators	Mandatory	None
AssociatorNames	Mandatory	None
References	Unspecified	None
ReferenceNames	Unspecified	None
EnumerateInstances	Mandatory	None
EnumerateInstanceNames	Mandatory	None

8.3 CIM_ReferencedProfile

Table 2 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 2 – Operations: CIM_ReferencedProfile

Operation	Requirement	Messages
GetInstance	Mandatory	None
ModifyInstance	Unspecified	None
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None
EnumerateInstances	Unspecified	None
EnumerateInstanceNames	Unspecified	None

8.4 CIM_ElementConformsToProfile

Table 3 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 3 – Operations: CIM_ElementConformsToProfile

Operation	Requirement	Messages
GetInstance	Mandatory	None
ModifyInstance	Unspecified	None
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None
EnumerateInstances	Unspecified	None
EnumerateInstanceNames	Unspecified	None

9 Use Cases (Informative)

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

9.1 Object Diagrams

Figure 4 represents an example instantiation of the *Profile Registration* in which both the Central Class methodology and the Scoping Class methodology for advertising profile compliance have been implemented. Additionally, the recommended use of an Interop Namespace and a separate Implementation Namespace is represented.

For simplicity, the prefix *CIM_* has been removed from the names of the classes.

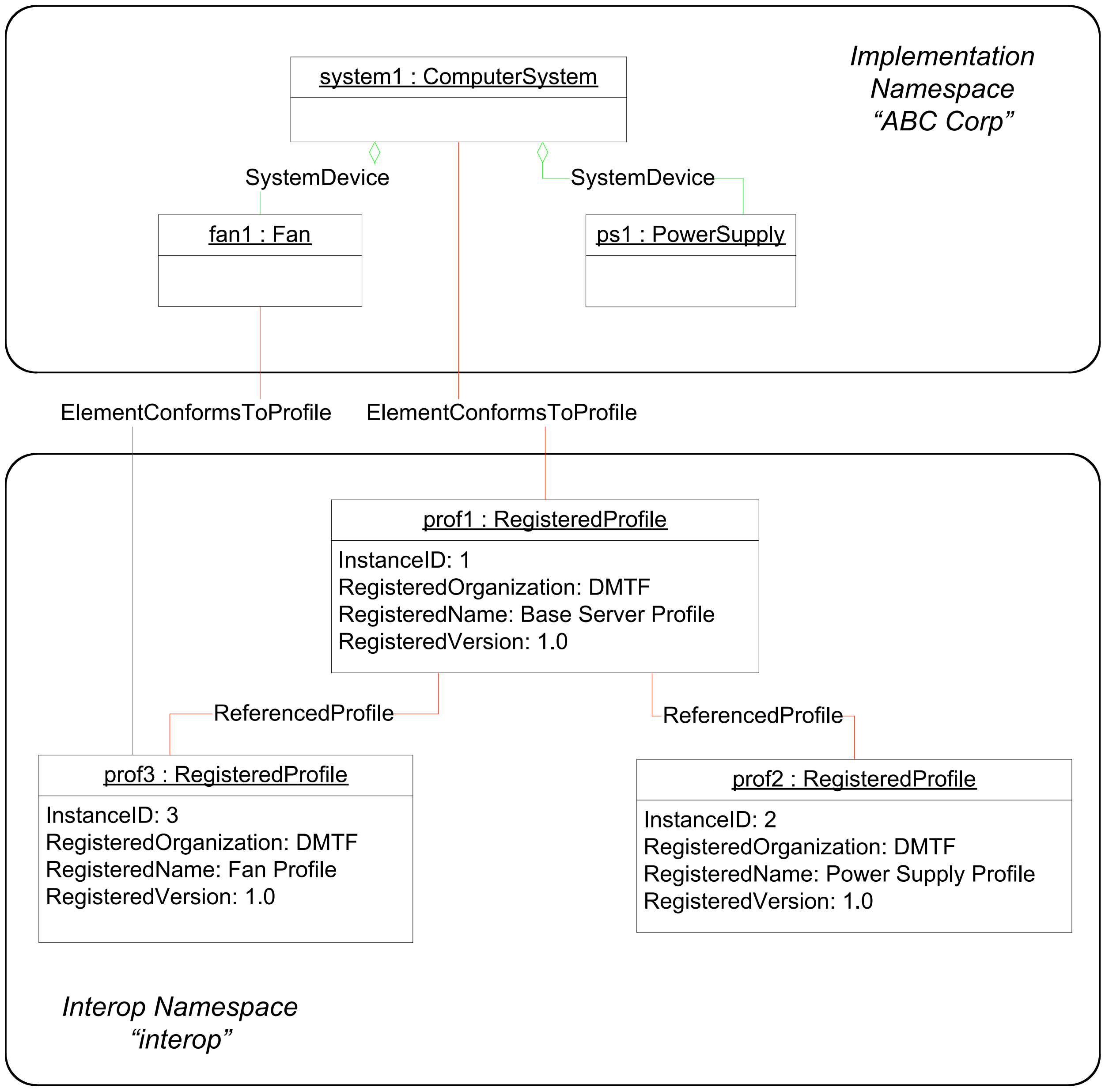

system1 : ComputerSystem
Implementation Namespace “ABC Corp”
SystemDevice
SystemDevice
fan1 : Fan
ps1 : PowerSupply
ElementConformsToProfile
ElementConformsToProfile
prof1 : RegisteredProfile
InstanceID: 1
RegisteredOrganization: DMTF
RegisteredName: Base Server Profile
RegisteredVersion: 1.0
ReferencedProfile
ReferencedProfile
prof3 : RegisteredProfile
InstanceID: 3
RegisteredOrganization: DMTF
RegisteredName: Fan Profile
RegisteredVersion: 1.0
prof2 : RegisteredProfile
InstanceID: 2
RegisteredOrganization: DMTF
RegisteredName: Power Supply Profile
RegisteredVersion: 1.0
Interop Namespace “interop”

Figure 4 – Profile Registration: Object Diagram

In Figure 4, the CIM_ComputerSystem instance system1 that represents the managed system is instantiated in the “ABC Corp” Implementation Namespace.

In the "ABC Corp" namespace, the instance of CIM_ElementConformsToProfile, which is connected to system1, contains the cross-namespace reference to the CIM_RegisteredProfile instance that contains information about the *Base Server Profile* (DSP1004) implementation in which system1 is the Scoping Instance. Additionally, the instance of CIM_ElementConformsToProfile that is connected to fan1 contains the cross-namespace reference to the CIM_RegisteredProfile instance, prof3, which contains information about the *Fan Profile* (DSP1013) implementation in which fan1 is the Central Instance. See 7.3 for more information about cross-namespace associations.

The *Base Server Profile* (DSP1004) includes references to the *Fan Profile* (DSP1013) and the *Power Supply Profile* (DSP1015) in its list of related profiles. Hence, an instance of the CIM_ReferencedProfile association exists between the CIM_RegisteredProfile instance for the *Base Server Profile* (DSP1004) and the CIM_RegisteredProfile instances for the *Fan Profile* (DSP1013) and the *Power Supply Profile* (DSP1015).

Determining profile implementation conformance for the ps1 instance must be done through the Scoping Class methodology because the instance has no direct CIM_ElementConformsToProfile association with the prof2 instance that represents the *Power Supply Profile* (DSP1015).

The Interop Namespace contains the instances of CIM_RegisteredProfile that identify the profile versioning information for all of the profiles that are implemented for the managed system.

Because CIM_ElementConformsToProfile instances are required to be instantiated in the Interop Namespace, a client of this profile can determine that “ABC Corp” is the name of the Implementation Namespace in which the profiles are implemented.

9.2 Retrieve the Profile Information for an Instance of CIM_ComputerSystem

Using Figure 4 and depending on the Central Class methodology for advertising profile conformance, a client may retrieve the profile information for an instance of CIM_ComputerSystem as follows:

1) Select the CIM_ComputerSystem instance in the Implementation Namespace.
2) Follow the CIM_ElementConformsToProfile association to the instance of CIM_RegisteredProfile.
3) Select the RegisteredOrganization, RegisteredName, and RegisteredVersion property values of the CIM_RegisteredProfile instance.

9.3 Retrieve the Profile Version Information for a Specific Fan

Using Figure 4, the following procedure describes the algorithm to retrieve the profile information for a fan where the provider has implemented the Central Class methodology for advertising profile compliance:

1) Find the CIM_ElementConformsToProfile cross-namespace association that is associated to the instance of CIM_Fan that represents the given fan.
2) Select the RegisteredOrganization, RegisteredName, and RegisteredVersion property values of the associated CIM_RegisteredProfile instance.

Figure 2 shows the object diagram for the procedure.

9.4 General Algorithm for Retrieving Profile Information

The following Storage Management Initiative Recipe Language (SMIRL) pseudo-code defines an algorithm that a client application could use to determine profile implementation information for an instance of CIM_Fan. This algorithm works regardless of whether the implementation is using the Central Class or Scoping Class methodology to advertise profile implementation conformance. The instance of CIM_Fan could be any subclass of CIM_LogicalDevice that has been identified as a Central Class in a profile. Other ManagedElements, which are not subclasses of CIM_LogicalDevice, that are the Central Class of a profile can be found with this algorithm by using a different association between the Central Class and Scoping Class as defined in the profile. Profile RegisteredName, RegisteredVersion, and RegisteredOrganization information can be retrieved from the instance of CIM_RegisteredProfile yielded by this algorithm.

```
/* Find the Profile instance governing a particular instance of a CIM_Fan
 * class defined as the Central Class of the DMTF Fan Profile.
 * Preconditions - $fan identifies the fan about which we are interested
 * in finding related profiles.
 * Note: This algorithm is not applicable if the SMIS (Storage Management
 * Interface Specification) Multiple Computer System subprofile is
 * implemented.
 *
 * Assumptions   - CIM_Fan is defined by a profile named "Fan Profile".
 *               - DMTF Fan Profile defines CIM_Fan to be the Central Class.
 *               - DMTF Fan Profile defines CIM_ComputerSystem (a subclass
 *                   of CIM_System) as the Scoping Class.
 *               - Cross-namespace associations have been implemented to
 *                   support traversal between Interop and Implementation
 *                   Namespaces.
 *
 * Postconditions - $subjectProfile will contain the RegisteredProfile
 * instance of the DMTF Fan Profile.
*/
// Step 1) Check whether there exists an explicit
// ElementConformsToProfile to our target instance; this takes precedence
// over any scoping profile.
   $profiles->[] = Associators($fan.getObjectPath(),
                        "CIM_ElementConformsToProfile",
                        "CIM_RegisteredProfile",
                        null,
                        null,
                        false,
                        false,
                        null);
//none directly associated
if ($profiles->[].length == 0) {
   // Step 2) Find the scoping System instance (will work for subclasses
   // like CIM_ComputerSystem).
   $scoping->[] = Associators($fan.getObjectPath(),
                        "CIM_SystemDevice",
                        "CIM_System",
                        null,
                        null,
                        false,
                        false,
                        null);
   $sysInstance-> = $scoping->[0];
   // Step 3) Find the autonomous profile for the scoping System instance.
   $profiles->[] = Associators($sysInstance->,
                        "CIM_ElementConformsToProfile",
                        "CIM_RegisteredProfile",
                        null,
                        null,
                        false,
                        false,
                        null);
```

```
    // save reference to the Autonomous profile for the CS instance
    $autoRP-> = $profiles->[0];
    //now find the associated component profiles
    $profiles->[] = Associators($autoRP->,
                        "CIM_ReferencedProfile",
                        "CIM_RegisteredProfile",
                        "Antecedent",
                        "Dependent",
                        false,
                        false,
                        null);
    //look for fan profiles
    for (#i=0; #i<$profiles->[].length; $i++) {
        //it's a fan
        if ($profiles->[#i].RegisteredName == "Fan Profile") {
            $fanProfiles->[$fanProfiles->[].length] = $profiles->[#i];
        }
    }
    // Step 4) Find the scoping Fan Profile.
    for (#i=0; $fanProfiles->[].length; #i++) {
        $associated->[] = Associators($fanProfiles->[#i],
                        "CIM_ElementConformsToProfile",
                        "CIM_Fan",
                        null,
                        null,
                        false,
                        false,
                        null);
        if ($associated->[].length == 0) {
            //no explicit ElementConformsToProfile, use scoping profile
                $subjectProfile-> = $fanProfiles->[#i];
        }
    }
} else {
    //the defining profile is directly associated
    $subjectProfile-> = $profiles->[0];
}
```

9.5 Use an Association Path Traversal to Determine Conformance

This use case demonstrates a discovery methodology that a client may use to ascertain if an instance is conformant to a particular profile.

This discovery algorithm specifies how a client may traverse specific associations from the Central Instance to other instances through associations that are defined in the subject profile.

When conformance of an instance of a CIM_ManagedElement to a profile is determined using the Scoping Class methodology, the client discovery requires additional association traversal to the Scoping Class instance, and then traversal to the CIM_RegisteredProfile hierarchy that contains the profile.

To determine profile conformance where the Scoping Class methodology is used, see 9.4, "General Algorithm for Retrieving Profile Information."

Figure 5 illustrates a profile that contains multiple elements where a client would use the association path from the Central Instance to other elements defined in the profile to determine the conformance of the element.

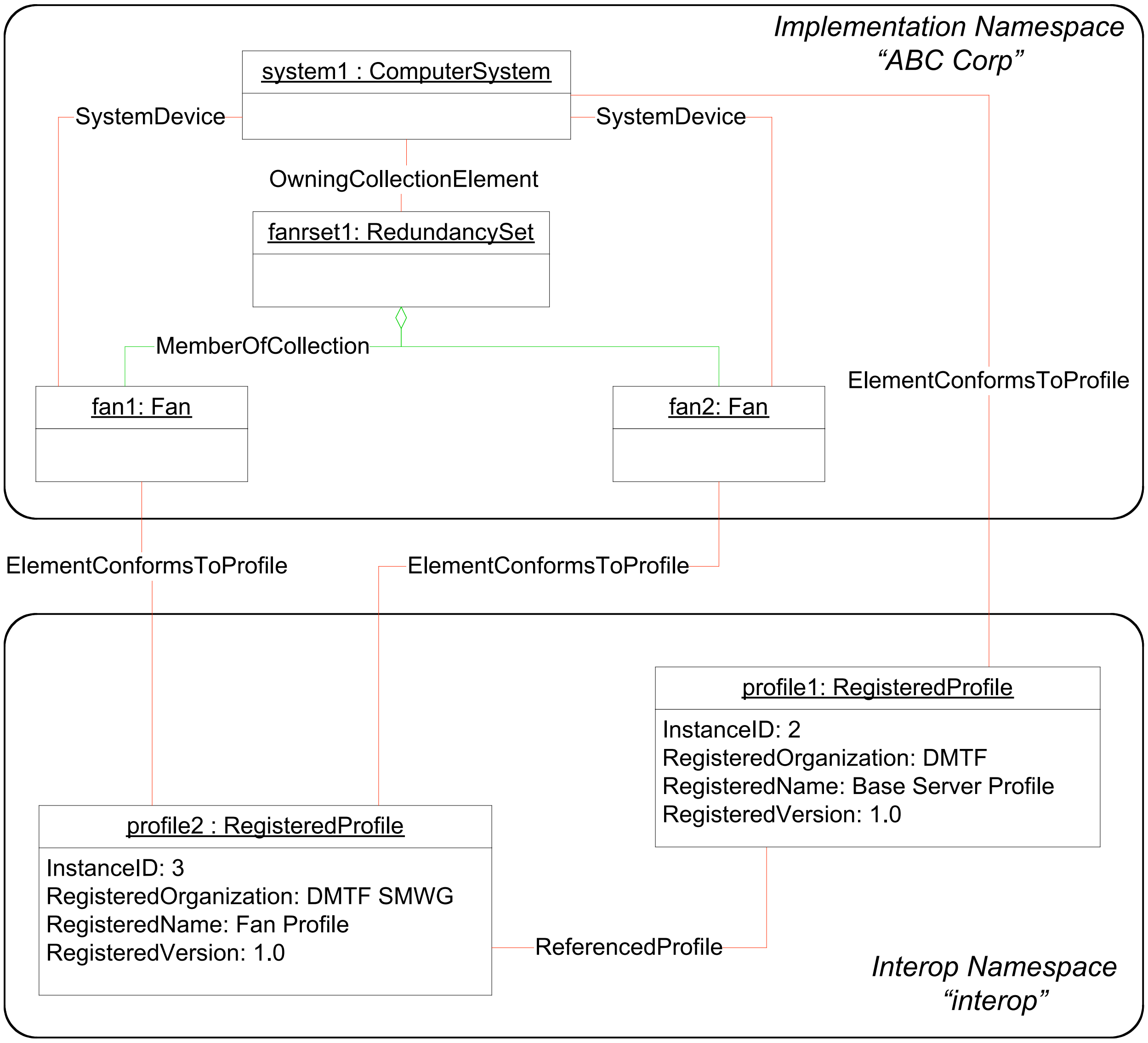


Figure 5 – Redundant Fan: Object Diagram

In Figure 5, CIM_Fan is the Central Class of the *Fan Profile* (DSP1013). Each instance of CIM_Fan is a Central Instance. The use of CIM_RedundancySet in this situation is defined in DSP1013, and the implementation of associated instances like these will be in conformance with the profile with which they are directly or indirectly associated. An association path exists to the CIM_RedundancySet from a Central Instance of DSP1013. Thus, a client can determine that fanrset1 is compliant with DSP1013.

9.6 Enumerate Profiles Advertised in Interop Namespace by an Implementation

The following SMIRL pseudo-code describes the algorithm for determining the profiles implemented and advertised in an Interop Namespace:

```
// DESCRIPTION
// A management application wishes to determine the profiles advertised
// in an Interop Namespace.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1. Assume the client has already determined and connected to the
// Interop Namespace
// Step 1: Get the names of all the RegisteredProfiles in the
// Interop Namespace
#ProfileName[] = EnumerateInstances("CIM_RegisteredProfile",
                 TRUE, TRUE, FALSE, FALSE,
                 ["RegisteredName"])
```

9.7 Determine Top-Level Profiles in an Interop Namespace

In an Interop Namespace, determining what "top-level" profiles are implemented is accomplished by determining which instances of CIM_RegisteredProfile are *not* antecedents for any CIM_ReferencedProfile associations. For this use case, top-level profiles are typically autonomous profiles that represent the largest scoping of the CIM representation of the target system. Top-level profiles may define as related profiles other autonomous and many component profiles. Examples of top-level profiles are a Base Server Profile that may reference Service Processor, Fan, and Power Supply Profiles, or a Modular System Profile that may reference Chassis Manager and Base Server Profiles.

The following SMIRL pseudo-code defines an algorithm for determining what top-level profiles are advertised in an Interop Namespace:

```
// DESCRIPTION
// A management application wishes to determine the profiles advertised
// in a particular namespace.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1. Assume the client has already determined and connected to the
// Interop Namespace
// Step 1: Get the instances of all the CIM_RegisteredProfiles in the
// Interop Namespace.
$AllProfiles[] = EnumerateInstances("CIM_RegisteredProfile",
                    TRUE, TRUE, FALSE, FALSE,
                    ["RegisteredName"])
// Step 2: Get the names of all the CIM_RegisteredProfiles in the
// Interop Namespace that are referenced as the Antecedent of any
// instances of CIM_ReferencedProfile.
$DependentProfileNames[] = AssociatorNames("CIM_RegisteredProfile",
                     "CIM_ReferencedProfile",
                     "CIM_RegisteredProfile",
                    Antecedent, NULL )
// Step 3: Subtract the DependentProfileNames list from the AllProfiles list
// by erasing any profile name in the AllProfilesNames list that is in
// the DependentProfileNames list.
for (#i=0; #i<$AllProfiles[].length; #i++) {
   for (#j=0; #j<$DependentProfileNames[].length; #j++) {
      if ($AllProfiles[#i].getObjectName() == $DependentProfileNames[#j]) {
      $AllProfiles[#i] = NULL; }
   }
}
// Step 4: AllProfiles is now a sparse array that contains only
// the instances of profiles that are top level.
```

9.8 Determine Implementation Instances for a Profile

The following SMIRL pseudo-code describes the algorithm for determining the Central Class implementation instances for a profile advertised in an Interop Namespace. This algorithm depends on the Central Class methodology for advertising profile implementation.

```
// DESCRIPTION
// A management application wishes to determine the ManagedElements that
// are instantiated by a particular DMTF profile, specifically the CPU
// Profile version 1.0.0.
//
// PRE-EXISTING CONDITIONS AND ASSUMPTION
// 1. Assume the client has located and connected to the Interop Namespace.
//
// Step 1: Select the instance of CIM_RegisteredProfile that represents
// the DMTF CPU Profile version 1.0.0.
$profiles->[] = EnumerateInstances("CIM_RegisteredProfile",
                     TRUE, TRUE, FALSE, FALSE,
                     ["RegisteredName"])
   for (#i=0; #i<$profiles->[].length; $i++) {
      if ($profiles->[#1].RegisteredOrganization = "DMTF") AND
         ($profiles->[#i].RegisteredName == "CPU Profile") AND
         {$profiles->[#1].RegisteredVersion == "1.0.0")
        then $RegisteredProfile-> = $profiles->[#i];
       }
// Step 2: If the $RegisteredProfile-> variable is null, then return an error
// as there are no instances of CIM_RegisteredProfile that represent the
// DMTF CPU Profile version 1.0.0
//
// Step 3: Determine the ManagedElement (System) by traversing the
// ElementConformsToProfile association from the RegisteredProfile
$ManagedElement->[] = Associators (
          $RegisteredProfile->,
          "CIM_ElementConformsToProfile",
          NULL,
          NULL,
          FALSE,
          NULL)
// Step 4: If the $ManagedElement->[] array has no elements, return an error
<ERROR! there are no instances of the Central Class implemented for this
```

```
  profile or the implementation has not utilized the Central Class
  methodology for advertising implementation conformance.>
//
// Step 5: The object name of more than one ManagedElement may be contained
// in the array returned. Examine the contents of $ManagedElement[]
// and save the name of the element of interest as $Name.
```

9.9 Peer Component Profile Relationships

Figure 6 illustrates the relationship between CIM_RegisteredProfile instances for the peer component profiles Fan and Sensor. The implementation and Interop Namespaces are depicted for illustrative purposes showing a typical implementation. See 7.1 for more information about namespaces.

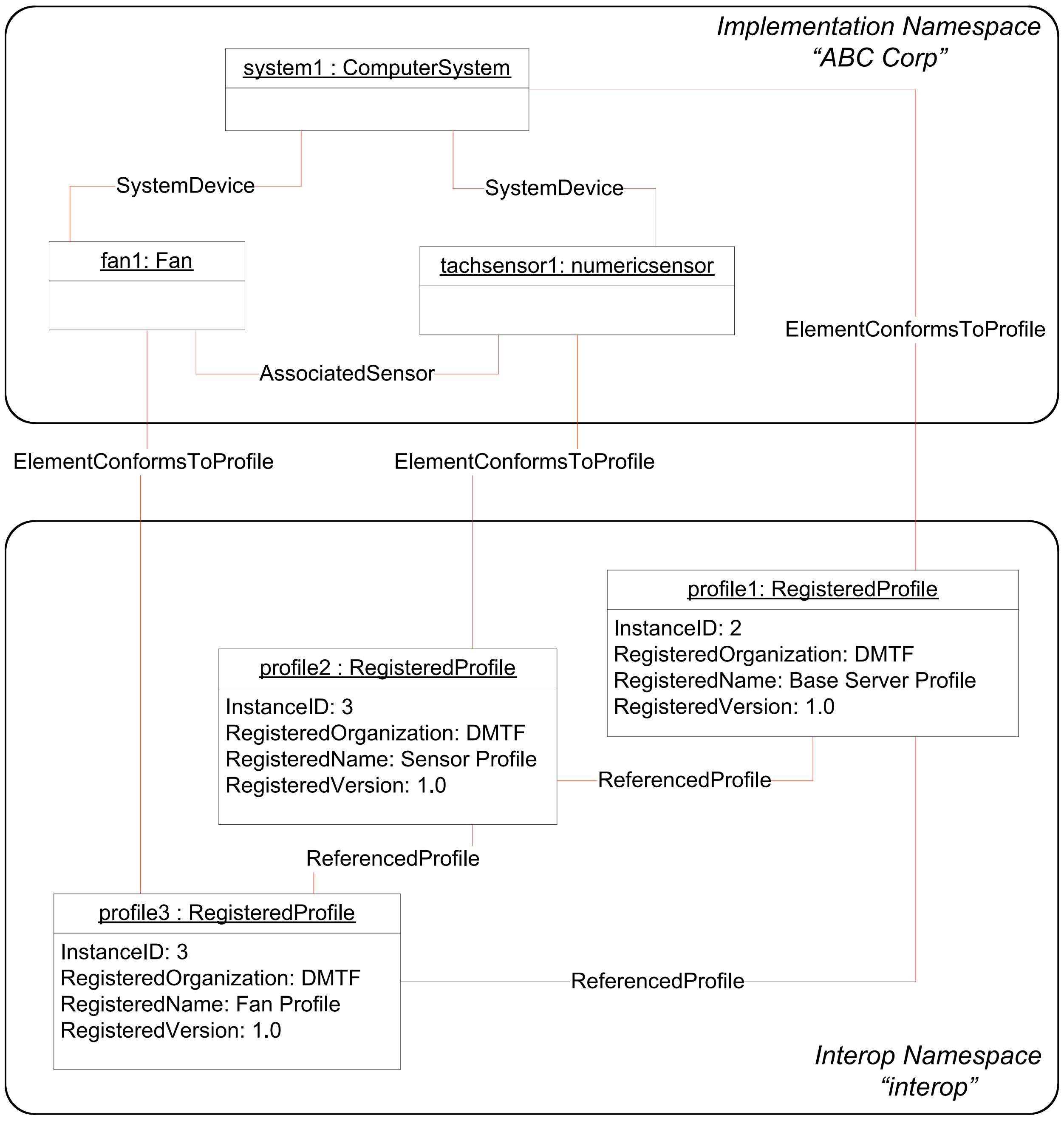


Figure 6 – Peer Component Profiles: Object Diagram

In Figure 6, the Central Instances of three profiles are shown associated through CIM_ElementConformsToProfile to the instances of CIM_RegisteredProfile that represent the profiles with which they are compliant. Also represented is the CIM_RegisteredProfile hierarchy through the CIM_ReferencedProfile associations in the Interop Namespace. In this situation, the *Base Server Profile* (DSP1004) is the autonomous profile that references the component profiles, the *Sensor Profile* (DSP1009) and the *Fan Profile* (DSP1013). This hierarchy would support the Scoping Class methodology for profile compliance advertisement. The relationship between peer component profiles, Fan and Sensor (that is, the *Fan Profile* includes the *Sensor Profile* and defines a tachometer sensor), is represented by an instance of the CIM_ReferencedProfile association.

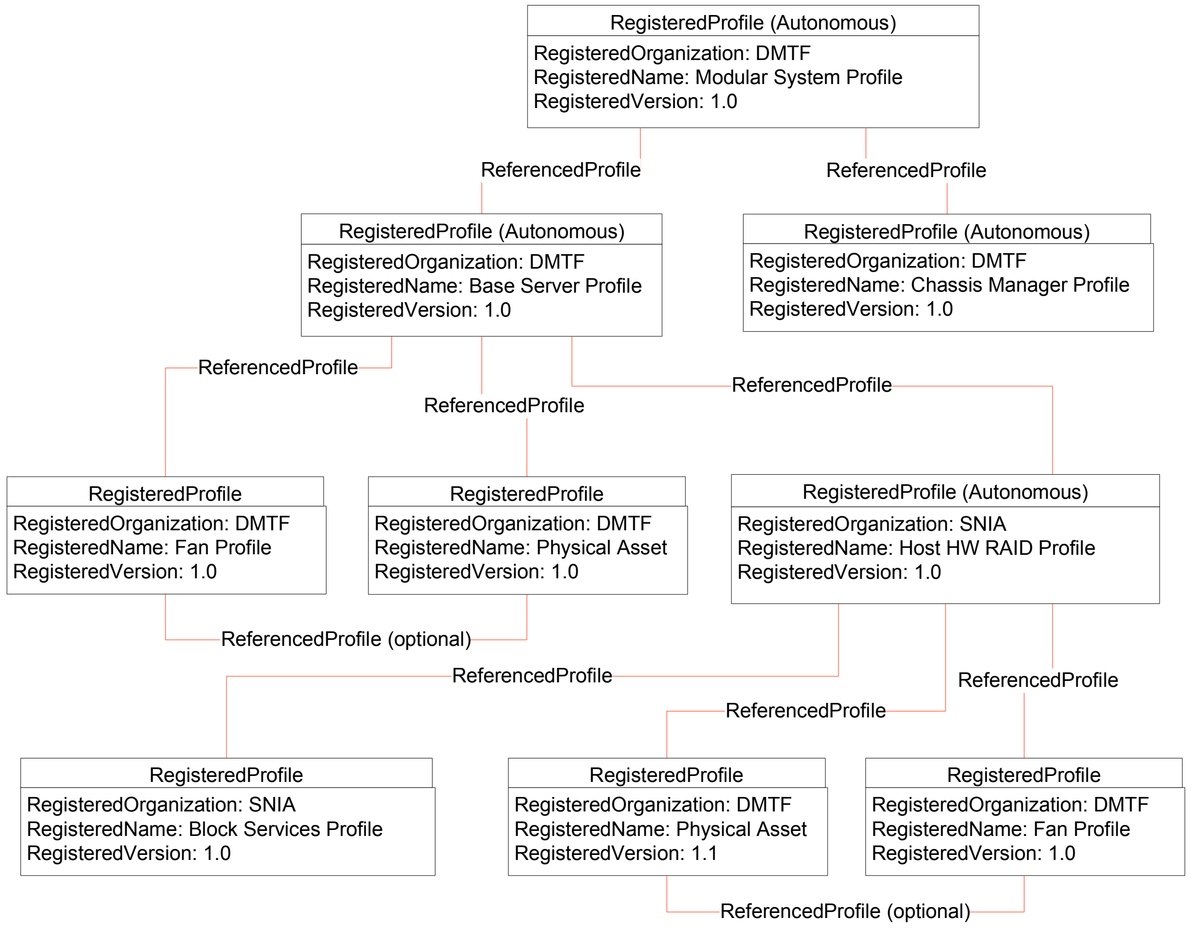
9.10 Determine Whether Central or Scoping Class Methodology Is in Use

For a specific instance of CIM_RegisteredProfile that represents the implementation advertisement of conformance to a profile, determining whether the Central Class methodology or the Scoping Class methodology has been used is based on whether CIM_ElementConformsToProfile associations are directly linked to the instance of CIM_RegisteredProfile in question.

- If one or more instances of CIM_ElementConformsToProfile are directly associated with the specific instance of CIM_RegisteredProfile, the Central Class methodology is being used.
- If the specific instance of CIM_RegisteredProfile is an autonomous profile and is the top-level profile in the CIM_RegisteredProfile hierarchy that comprehensively represents the system, both the Central Class and Scoping Class methodologies are being used. This situation is unique because the Central Class and the Scoping Class in this type of autonomous profile are the same and the Scoping Class methodology is based on the CIM_ElementConformsToProfile link between the top-level CIM_RegisteredProfile and the top-level managed element in the Implementation Namespace.

9.11 Example of Profile Compliance Hierarchy

Figure 7 depicts the hierarchy of instances of CIM_RegisteredProfile associated through instances of CIM_ReferencedProfile that would represent a modular system with a chassis manager and an included blade server with RAID storage. Figure 7 is provided as an example to illustrate the nature of the relationships among the various autonomous and component profiles. Also depicted are the relationships between component profiles.


RegisteredProfile (Autonomous)
RegisteredOrganization: DMTF
RegisteredName: Modular System Profile
RegisteredVersion: 1.0
ReferencedProfile
ReferencedProfile
RegisteredProfile (Autonomous)
RegisteredOrganization: DMTF
RegisteredName: Base Server Profile
RegisteredVersion: 1.0
RegisteredProfile (Autonomous)
RegisteredOrganization: DMTF
RegisteredName: Chassis Manager Profile
RegisteredVersion: 1.0
ReferencedProfile
ReferencedProfile
ReferencedProfile
RegisteredProfile
RegisteredOrganization: DMTF
RegisteredName: Fan Profile
RegisteredVersion: 1.0
RegisteredProfile
RegisteredOrganization: DMTF
RegisteredName: Physical Asset
RegisteredVersion: 1.0
RegisteredProfile (Autonomous)
RegisteredOrganization: SNIA
RegisteredName: Host HW RAID Profile
RegisteredVersion: 1.0
ReferencedProfile (optional)
ReferencedProfile
ReferencedProfile
ReferencedProfile
RegisteredProfile
RegisteredOrganization: SNIA
RegisteredName: Block Services Profile
RegisteredVersion: 1.0
RegisteredProfile
RegisteredOrganization: DMTF
RegisteredName: Physical Asset
RegisteredVersion: 1.1
RegisteredProfile
RegisteredOrganization: DMTF
RegisteredName: Fan Profile
RegisteredVersion: 1.0
ReferencedProfile (optional)

864

865 **Figure 7 – Profile Compliance Hierarchy**

10 CIM Elements

Table 4 shows the instances of CIM Elements for this profile. Instances of the CIM Elements shall be implemented as described in Table 4. Sections 7 (“Implementation”) and 8 (“Methods”) may impose additional requirements on these elements.

Table 4 – CIM Elements: Profile Registration Profile

Element Name	Requirement	Description
Classes		
CIM_RegisteredProfile	Mandatory	Shall be in the Interop Namespace and may be instantiated in any other namespace
CIM_ElementConformsToProfile	Mandatory	Shall be instantiated in the Interop Namespace and should be instantiated in the Implementation Namespace if it is separate from the Interop Namespace
CIM_ReferencedProfile	Mandatory	Shall be in the Interop Namespace and may be instantiated in other namespaces See 7.3 for more information about cross-namespace associations.
Indications		
None defined in this profile		

10.1 CIM_RegisteredProfile

CIM_RegisteredProfile is used to advertise implementation conformance to a CIM model profile. Table 5 defines the requirements for elements of this class.

Table 5 – Class: CIM_RegisteredProfile

Elements	Requirement	Notes
InstanceID	Mandatory	Key
RegisteredOrganization	Mandatory	See 7.9.1.
ReqisteredName	Mandatory	See 7.9.2.
RegisteredVersion	Mandatory	See 7.9.3.
AdvertiseTypes	Mandatory	Required qualifier
OtherRegisteredOrganization	Conditional	Mandatory if RegisteredOrganization contains the value “Other”
AdvertiseTypeDescriptions	Conditional	Mandatory if AdvertiseTypes contains the value “Other”

10.2 CIM_ElementConformsToProfile

CIM_ElementConformsToProfile is used to associate an instance of a subclass of CIM_ManagedElement with a corresponding instance of CIM_RegisteredProfile to which the managed element belongs. Table 6 defines the requirements for elements of this class.

Table 6 – Class: CIM_ElementConformsToProfile

Elements	Requirement	Notes
ConformantStandard	Mandatory	Key: REF to the instance of CIM_RegisteredProfile
ManagedElement	Mandatory	Key: REF to the instance of a subclass of CIM_ManagedElement

10.3 CIM_ReferencedProfile

CIM_ReferencedProfile is used to associate an instance CIM_RegisteredProfile with an instance of CIM_RegisteredProfile of another profile that references the dependent profile as a related profile. Table 7 defines the requirements for elements of this class.

Table 7 – Class: CIM_ReferencedProfile

Elements	Requirement	Notes
Antecedent	Mandatory	Key: See 7.11.1.
Dependent	Mandatory	Key: See 7.11.2.

ANNEX A
(informative)

Change Log

Version	Date	Description
1.0.0 Preliminary	2006/12/06	Preliminary standard.
1.0.0 Final	2007/06/25	Final version

.

ANNEX B
(informative)

Acknowledgements

The authors wish to acknowledge the following people.

Editor:

- Jon Hass – Dell

Contributors:

- Aaron Merkin – IBM
- Steve Hand - Symantec
- Khachatur Papanyan – Dell
- Christina Shaw – HP
- John Leung – Intel
- Mike Walker – IBM
- Paul von Behren – Symantec
- Jim Davis – WBEM Solutions
- George Ericson – EMC