**Document Number: DSP1023**

**Date: 2008-09-04**

**Version: 1.0.0**

# Software Inventory Profile

**Document Type: Specification**

**Document Status: Final Standard**

**Document Language: E**

# CONTENTS

## Figures

## Tables

# Foreword

The *Software Inventory Profile* (DSP1023) was prepared by the Server Management Working Group.

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

# Introduction

The information in this specification should be sufficient for a provider or consumer of this data to identify unambiguously the classes, properties, methods, and values that are instantiated and manipulated to identify and query the inventory of installed BIOS, firmware, drivers, and related software in a managed system. This profile also describes the Common Information Model (CIM) schema elements required to represent the software that can be installed on a managed system.

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

# Software Inventory Profile

## 1 Scope

The *Software Inventory Profile* describes the CIM schema elements required to provide an inventory of installed BIOS, firmware, drivers, and related software in a managed system. This profile also describes the CIM schema elements required to represent the software that can be installed on a managed system.

## 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 DSP0200, *CIM Operations over HTTP 1.2.0*

DMTF DSP0004, *CIM Infrastructure Specification 2.3.0*

DMTF DSP1000, *Management Profile Specification Template 1.0.0*

DMTF DSP1001, *Management Profile Specification Usage Guide 1.0.0*

DMTF DSP1033, *Profile Registration Profile 1.0.0*

### 2.2 References under Development

DMTF DSP1025, *Software Update Profile 1.0.0*

### 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)*

## 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 DSP1033 and 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**

**Software Identity**

an instance of CIM_SoftwareIdentity that represents and contains the identifying property values of a software image

**3.15**

**Installed Software**

software that is installed on any managed element in the scope of a system

**3.16**

**Available Software**

software that the management infrastructure has determined is available, either locally or at a remote location, for installation on the managed system and may be appropriate to install without any assertion about the ability to perform the installation through the management infrastructure

**3.17**

**Software Bundle**

a software image that consists of one or more discrete software images that can be installed individually or together

**3.18**

**Managed Element**

an instance of CIM_ManagedElement that represents a managed element in the scope of a system

**3.19**

**Software Family**

a group of software in which each member software could be installed in the place of the other on a Managed Element and offer similar functionality to a Managed Element

**3.20**

**Installation Dependency**

a software image that needs to be installed before installing the target Software Identity

# 4 Symbols and Abbreviated Terms

**Experimental Maturity Level**

Some of the content considered for inclusion in *Software Inventory Profile* has yet to receive sufficient review to satisfy the adoption requirements set forth by the Technical Committee within the DMTF. This content is presented here as an aid to implementers who are interested in likely future developments within this specification. The content marked experimental may change as implementation experience is gained. There is a high likelihood that it will be included in an upcoming revision of the specification.  Until that time, it is purely informational, and is clearly marked within the text.

A sample of the typographical convention for experimental content is included here:

**EXPERIMENTAL**

Experimental content appears here

**EXPERIMENTAL**

The following symbols and abbreviations are used in this document.

None

# 5 Synopsis

**Profile Name:** Software Inventory

**Version:** 1.0.0

**Organization:** DMTF

**CIM Schema version:** 2.20

**Central Class:** CIM_SoftwareIdentity

**Scoping Class:** CIM_System

The *Software Inventory Profile* describes the classes and properties used to provide an inventory of installed BIOS, firmware, drivers, and related software in a managed system. This profile also describes the classes and properties required to represent the software that can be installed on a managed system. The profile defines the use of a Software Identity for representing the software image known to the managed system. The profile also defines the relationship between a Managed Element and the Software Identity that is applicable to that Managed Element.

CIM_SoftwareIdentity shall be the Central Class of this profile. The instance of CIM_SoftwareIdentity shall be the Central Instance of this profile.

CIM_System shall be the Scoping Class of this profile. The instance of CIM_System shall be the Scoping Instance of this profile and shall be selected using the algorithm described in section 7.5.

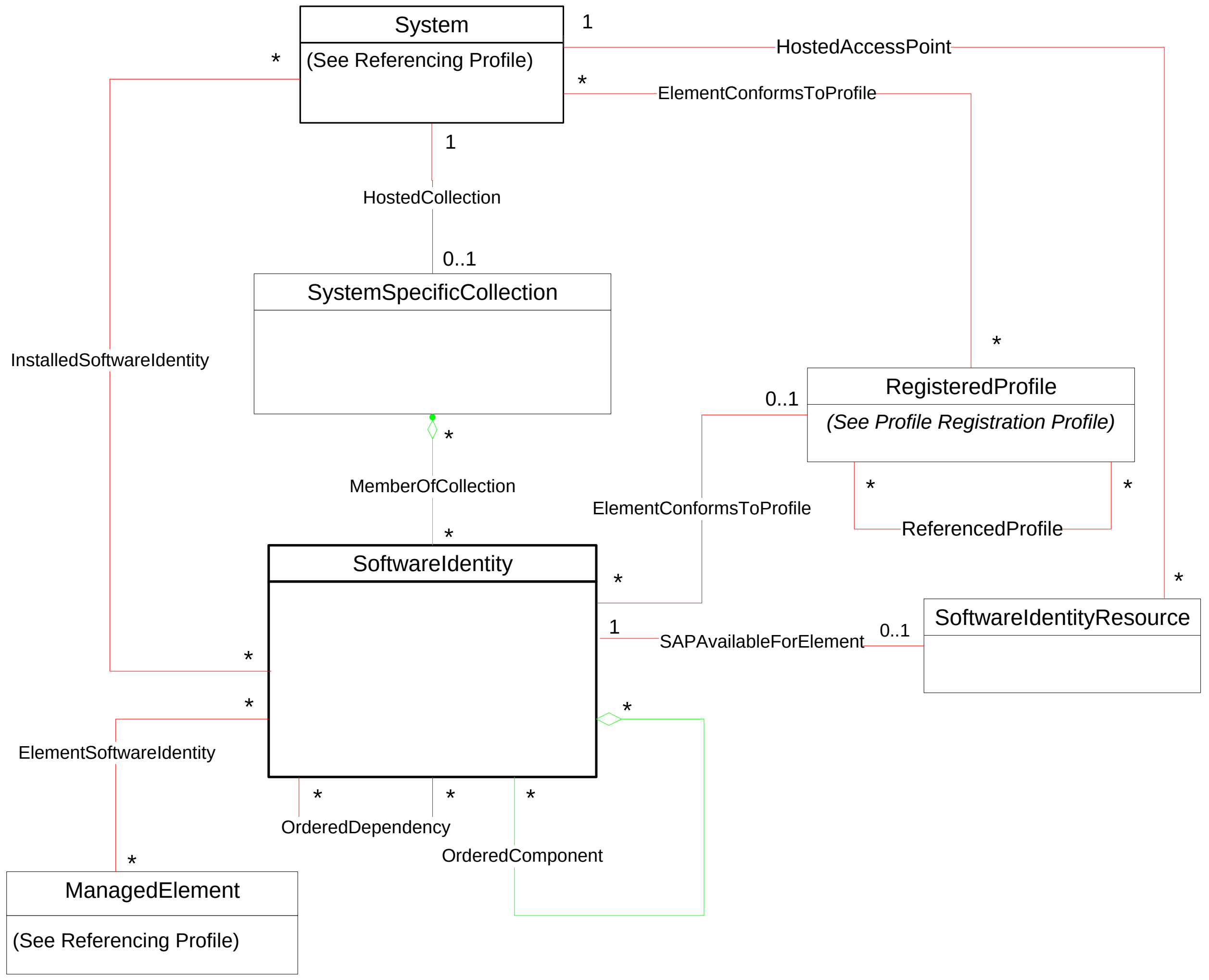
References to CIM_System may be interpreted as references to subclasses of CIM_System such as CIM_ComputerSystem. Table 1 identifies profiles on which this profile has a dependency.

**Table 1 – Referenced Profiles**

| Profile Name | Organization | Version | Description |
|---|---|---|---|
| Profile Registration | DMTF | 1.0 | Mandatory |

# 6 Description

The *Software Inventory Profile* provides the ability to perform an inventory of installed BIOS, firmware, drivers, and related software such as providers and instrumentation software. This profile also describes the CIM schema elements required to represent the software that can be installed on a managed system. It also provides information about what software is associated with particular Managed Elements such as devices. Figure 1 represents the class schema of the *Software Inventory Profile* and shows the elements of the *Software Inventory Profile*, as well as the dependent relationships between the elements of *Software Inventory Profile* and the referencing profiles. For simplicity, the prefix *CIM_* has been removed from the names of the classes.



292

293 **Figure 1 – Class Diagram: Software Inventory Profile**

294 The *Software Inventory Profile* can be used to represent the following software:

- 295 296 297 the software that is installed on any Managed Element in the scope of the managed system (see section 7.2) so that the user of the profile can inventory the installed software for the managed system
- 298 299 300 the software that is available for installation on any Managed Element in the scope of the managed system (see section 7.6) for providing the user of the profile the capability to view all the software that is available for any Managed Element within the scope of the managed system

301 302 For Available or Installed Software, the relationship between a Managed Element and the software that is compatible with the Managed Element (see section 7.4) can be modeled.

# 7 Implementation

This section describes the implementation requirements of the *Software Inventory Profile*. Required methods are described in section 8 (“Methods”), and properties are described in section 10 (“CIM Elements”).

## 7.1 Representing Software

The implementation shall model Installed Software (see section 7.2), Available Software (see section 7.6), or both, as a part of this profile.

## 7.2 Representing Installed Software

When an implementation models Installed Software, each Installed Software image modeled by the implementation shall be represented by exactly one instance of CIM_SoftwareIdentity. The IsEntity property of the instance of CIM_SoftwareIdentity shall have the value true.

### 7.2.1 CIM_InstalledSoftwareIdentity Instance

The Software Identity that represents an Installed Software shall be associated to the Scoping Instance using exactly one instance of CIM_InstalledSoftwareIdentity.

## 7.3 Representing Version Information of Software

When the version information is not represented using the VersionString property, it shall be represented using the MajorVersion, MinorVersion, RevisionNumber, and BuildNumber properties. These properties are conditional and shall be implemented when the VersionString property is Null. When MinorVersion has a non-Null value, MajorVersion shall have a non-Null value. When RevisionNumber has a non-Null value, MinorVersion shall have a non-Null value. When BuildNumber has a non-Null value, RevisionNumber shall have a non-Null value. The algorithm for comparing versions of two instances of CIM_SoftwareIdentity using these properties is described in section 7.10.

## 7.4 Representing Relationships between Software Identity and Managed Element

The relationships between the software and the Managed Element may be modeled. This behavior is optional. When this behavior is implemented, the requirements specified in the following sections shall be met.

### 7.4.1 CIM_ElementSoftwareIdentity Instance

When a Managed Element is represented, the relationships between the Managed Element and the compatible Software Identity shall be represented using an instance of CIM_ElementSoftwareIdentity.

When the Managed Element is not represented with an instance, the relationship between the compatible Software Identity and the Managed Element may be still represented by associating the Software Identity to the Scoping Instance through an instance of CIM_ElementSoftwareIdentity.

#### 7.4.1.1 CIM_ElementSoftwareIdentity.ElementSoftwareStatus

The CIM_ElementSoftwareIdentity.ElementSoftwareStatus property shall represent the relationships of the software, represented by the Software Identity, to the Managed Element, through one or more enumeration values.

If the relationship between the Software Identity and the Managed Element is unknown, then the CIM_ElementSoftwareIdentity.ElementSoftwareStatus property shall contain no enumeration values.

NOTE: The ElementSoftwareStatus property does not convey the current status of the Managed Element itself.

#### 7.4.1.1.1 CIM_ElementSoftwareIdentity.ElementSoftwareStatus Enumeration Relationships

The relationships between the ElementSoftwareStatus property enumeration values on a single instance of CIM_ElementSoftwareIdentity are described in Table 2. When the ElementSoftwareStatus property of an instance of CIM_ElementSoftwareIdentity has the value specified in the “Enumeration Value” column of Table 2, the ElementSoftwareStatus property of the same instance shall also have other enumeration values specified in the corresponding row of the “Mandatory Pairing With” column of Table 2.

When the ElementSoftwareStatus property of an instance of CIM_ElementSoftwareIdentity has the value specified in the “Enumeration Value” column of Table 2, the ElementSoftwareStatus property of the same instance may also have other enumeration values specified in the corresponding row of the “May Be Used With” column of Table 2.

When the ElementSoftwareStatus property of an instance of CIM_ElementSoftwareIdentity has the value specified in the “Enumeration Value” column of Table 2, the ElementSoftwareStatus property of the same instance shall not have other enumeration values specified in the corresponding row of the “Shall Not Be Used With” column of Table 2.

NOTE: The "May Be Used With," "Mandatory Pairing With," and "Shall Not Be Used With" columns express the relationship of a contained value to the value in the "Enumeration Value" column. They do not express the relationship between two values contained in the column itself. Therefore, the occurrence of two values together in the "May Be Used With" column has no bearing on whether the two values may be used together.

**Table 2 – Relationships Between Enumeration Values of ElementSoftwareStatus**

| Enumeration Value | Mandatory Pairing With | May Be Used With | Shall Not Be Used With |
|---|---|---|---|
| 2 (Current) | | 3 (Next), 4 (FallBack), 5 (Default), 6 (Installed), 7 (SingleUse), 8 (Available) | 9 (Supports) |
| 3 (Next) | 6 (Installed) | 2 (Current), 4 (FallBack), 5 (Default) | 7 (SingleUse), 8 (Available), 9 (Supports) |
| 4 (FallBack) | 6 (Installed) | 2 (Current), 3 (Next), 5 (Default), 7 (SingleUse) | 8 (Available), 9 (Supports) |
| 5 (Default) | | 2 (Current), 3 (Next), 4 (FallBack), 6 (Installed), 7 (SingleUse), 8 (Available), 9 (Supports) | |
| 6 (Installed) | | 2 (Current), 3 (Next), 4 (FallBack), 5 (Default), 7 (SingleUse) | 8 (Available), 9 (Supports) |
| 7 (SingleUse) | 6 (Installed) | 5 (Default), 2 (Current), 4 (FallBack) | 3 (Next), 8 (Available), 9 (Supports) |
| 8 (Available) | | 2 (Current), 5 (Default) | 3 (Next), 4 (FallBack), 6 (Installed), 7 (SingleUse), 9 (Supports) |
| 9 (Supports) | | 5 (Default) | 2 (Current), 3 (Next), 4 (FallBack), 6 (Installed), 7 (SingleUse), 8 (Available) |

When a Software Identity that is associated with a Managed Element through an instance of CIM_ElementSoftwareIdentity with the ElementSoftwareStatus property containing the value 3 (Next) or 7 (SingleUse) fails to run, the system shall automatically attempt to use the Software Identity that is associated with the same Managed Element through an instance of CIM_ElementSoftwareIdentity with the ElementSoftwareStatus property containing the value 4 (FallBack), and no client action shall be required.

### 7.4.2 ElementSoftwareIdentity for Software That Is Intended for a Managed Element But Does Not Run or Get installed on It

When an instance of CIM_ElementSoftwareIdentity is used to represent the relationship between a Software Identity and a Managed Element such that the Software Identity will work with or can operate the Managed Element but is installed and runs on a different Managed Element, the only value that the ElementSoftwareIdentity.ElementSoftwareStatus property shall have is 9 (Supports).

## 7.5 Finding the Scoping Instance of the CIM_System Class

The following algorithm shall be used for locating the Scoping Instance of the CIM_System class from any instance of CIM_SoftwareIdentity:

1) If the selected instance is referenced by an instance of CIM_InstalledSoftwareIdentity, the Scoping Instance shall be the instance of CIM_System that is associated through the instance of CIM_InstalledSoftwareIdentity.

2) Otherwise, if the selected instance is referenced by the instance of CIM_MemberOfCollection, select the instance of CIM_SystemSpecificCollection that is associated through the instance of CIM_MemberOfCollection. The Scoping Instance of the profile shall be the instance of CIM_System that is associated with the selected instance of CIM_SystemSpecificCollection through the instance of CIM_HostedCollection.

## 7.6 Representing Available Software

When an implementation represents the Installed Software with Available Software, each Available Software image modeled by the implementation shall be represented by a Software Identity. The IsEntity property of the instance of CIM_SoftwareIdentity shall have the value true. The following subsections are applicable when Available Software is represented.

### 7.6.1 CIM_SystemSpecificCollection Instance

An implementation shall instantiate a single instance of CIM_SystemSpecificCollection, which is a collection of all the Available Software. The ElementName property of this instance of CIM_SystemSpecificCollection shall have a value of “Available Software”.

### 7.6.2 CIM_HostedCollection Instance

The instance of CIM_SystemSpecificCollection shall be associated to the Scoping Instance by exactly one instance of CIM_HostedCollection.

### 7.6.3 CIM_MemberOfCollection Instance

For each Software Identity that represents an Available Software, exactly one instance of CIM_MemberOfCollection shall associate the Software Identity to the CIM_SystemSpecificCollection instance.

**EXPERIMENTAL**

### 7.6.4 Advertising the Location Information of a Software Identity

The location of Available Software may be modeled. This behavior is optional. When this behavior is implemented, the requirements specified in the following sections shall be met.

#### 7.6.4.1 CIM_SoftwareIdentityResource Instance

The location of a Software Identity shall be represented by an instance of CIM_SoftwareIdentityResource. This could be used as an input to the software installation service. See the “Implementation” section of the *Software Update Profile*.

#### 7.6.4.2 CIM_SAPAvailableForElement Instance

An instance of CIM_SAPAvailableForElement shall be used to associate a Software Identity with a CIM_SoftwareIdentityResource instance that represents the location information of the Software Identity.

#### 7.6.4.3 CIM_HostedAccessPoint

An instance of CIM_HostedAccessPoint shall be used to associate a CIM_SoftwareIdentityResource instance and the CIM_System or CIM_ComputerSystem instance that represents the Scoping Instance of the Available Software whose location information is advertised by the CIM_SoftwareIdentityResource instance.

**EXPERIMENTAL**

### 7.6.5 Identifying Target Operating Systems

The operating systems supported by a Software Identity may be modeled. This behavior is optional. When this behavior is implemented, the target operating systems of a Software Identity shall be represented by using one or all of the methods described in the following sections.

#### 7.6.5.1 CIM_SoftwareIdentity.TargetOSTypes[]

The TargetOSTypes[] array property shall be used to list the operating systems that are supported by the Software Identity. An empty array shall indicate that the supported operating systems are unknown. A value of 66 (Not Applicable) shall indicate that the operating system is irrelevant when determining the compatibility of the Software Identity.

#### 7.6.5.2 CIM_SoftwareIdentity.TargetOperatingSystems[]

This TargetOperatingSystems[] property shall be used to represent the operating systems supported by the Software Identity that are not listed in the TargetOSTypes[] property array values.

**EXPERIMENTAL**

## 7.7 Representing a Software Bundle

A Software Bundle may be modeled. This behavior is optional. A Software Bundle shall be represented using a Software Identity. The Software Identity shall have a value of 13 (Software Bundle) in the Classifications[] property. Each software image in the Software Bundle shall be represented by a Software Identity that shall be associated to the Software Identity that represents the Software Bundle, using a single instance of CIM_OrderedComponent.

### 7.7.1 CIM_OrderedComponent.GroupComponent

The instance of CIM_SoftwareIdentity that represents the Software Bundle shall be the value of the GroupComponent property.

### 7.7.2 CIM_OrderedComponent.PartComponent

The instance of CIM_SoftwareIdentity that represents the individual software image that is a part of the Software Bundle shall be the value of the PartComponent property.

### 7.7.3 CIM_OrderedComponent.AssignedSequence

The AssignedSequence property indicates the order in which the Software Identity referenced by the CIM_OrderedComponent instance shall be installed during the installation of the bundle. The Software Identity with the lowest value of AssignedSequence on the associated CIM_OrderedComponent instance shall be installed first and the highest shall be installed last. An AssignedSequence value of zero shall indicate no ordering requirement. Equivalent values of the AssignedSequence property shall indicate no ordering preference.

## 7.8 Identifying a Software Identity

This section describes the use of the IdentityInfoType[] and IdentityInfoValue[] array properties to identify a Software Identity.

### 7.8.1 General Use of IdentityInfoType and IdentityInfoValue Properties

The IdentityInfoValue[] array property contains values that provide additional information to identify a Software Identity. The corresponding element in the IdentityInfoType[] array property shall indicate the type of information stored in the IdentityInfoValue[] array.

### 7.8.2 Using IdentityInfoType and IdentityInfoValue to Model a Software Family

Software Family is an application-specific invariant identifier that is consistent among versions of a Software Identity. Software Family may be used to correlate instances of the same software across namespaces or management infrastructures, regardless of version.

A Software Identity may belong to multiple Software Families. Each Software Family of the Software Identity shall be represented as follows:

- The IdentityInfoType[] array property shall have the value of “CIM:SoftwareFamily”.
- The corresponding element in the IdentifyingInfoValue[] array property shall be of the format “<OrgID> : <LocalID>”. <OrgID> shall include a copyrighted, trademarked, or otherwise unique name that is owned by the business entity creating or defining the Software Identity and LocalID is a unique value that is consistent among different versions of the software. The algorithm used to guarantee uniqueness of the LocalID is implementation specific. Two possible algorithms are as follows:

  1) Following is an example algorithm that may be used to generate the LocalID of a Software Identity for which the supported operating systems can be determined by the instrumentation:

     <CIM_SoftwareIdentity.Classifications[]>:<CIM_SoftwareIdentity.TargetOSTypes[]>:< Information of the Hardware/ Hardware family supported by the Software Identity>

     <CIM_SoftwareIdentity.Classifications[]> is one of the numeric values contained in the Classifications property, and <CIM_SoftwareIdentity.TargetOSTypes[]> is one of the values contained in the TargetOSTypes property of the instance of CIM_SoftwareIdentity.

  2) Following is an example algorithm that may be used to generate the LocalID of a Software Identity for which the supported operating systems cannot be determined by the instrumentation:

     <CIM_SoftwareIdentity.Classifications[]>:< Information of the Hardware/ Hardware family supported by the Software>

<CIM_SoftwareIdentity.Classifications[]> is one of the numeric values contained in the Classifications property of the instance of CIM_SoftwareIdentity.

#### 7.8.2.1 Determining Common Software Family Membership

Two instances of CIM_SoftwareIdentity shall belong to the same Software Family when at least one of the Software Families modeled for the first CIM_SoftwareIdentity instance matches at least one of the Software Families modeled for the second CIM_SoftwareIdentity instance.

**EXPERIMENTAL**

**EXPERIMENTAL**

## 7.9 Representing Installation Dependencies

Software on which a Software Identity is dependent may be modeled. This behavior is optional. When information about the dependency is known but a copy of the software is not modeled, the dependency shall be modeled using an instance of CIM_SoftwareIdentity and the IsEntity property shall have the value false. When information about the dependency is known and a copy of the software is modeled, the dependency shall be modeled using an instance of CIM_SoftwareIdentity and the IsEntity property shall have the value true.

### 7.9.1 CIM_OrderedDependency

When a Software Identity that is a member of the Available Software collection has installation dependencies on software that is represented by an instance of CIM_SoftwareIdentity, the instrumentation shall instantiate an instance of the CIM_OrderedDependency association between the Software Identity and each Installation Dependency, represented by an instance of CIM_SoftwareIdentity, to arrange the Installation Dependencies in a hierarchical order.

#### 7.9.1.1 CIM_OrderedDependency.Antecedent

The instance of CIM_SoftwareIdentity that represents the Installation Dependency shall be the value of the Antecedent property.

#### 7.9.1.2 CIM_OrderedDependency.Dependent

The instance of CIM_SoftwareIdentity for which the Installation Dependencies are represented shall be the value of the Dependent property.

#### 7.9.1.3 CIM_OrderedDependency.AssignedSequence

The AssignedSequence property indicates the order or sequence in which the Installation Dependencies shall be resolved during the installation of the Software Identity. The Installation Dependency with the lowest value of AssignedSequence on the associated CIM_OrderedComponent instance shall be installed first and the highest shall be installed last. An AssignedSequence value of zero shall indicate no ordering requirement.

**EXPERIMENTAL**

### 7.10 Version Comparison Using the MajorVersion, MinorVersion, RevisionNumber, and BuildNumber Properties

The following algorithm shall be used to indicate that a CIM_SoftwareIdentity instance has a higher version than the other instance of CIM_SoftwareIdentity when two instances of CIM_SoftwareIdentity are compared.

When comparing two properties in each of the following steps, if only one of the properties is Null, the instance that has a non-Null property shall be the instance with the higher version. When both properties are Null, the two instances shall be considered as having equal value.

1) If the MajorVersion properties of the two instances are equal, go to step 2.

   Otherwise, the instance with the higher value of the MajorVersion property shall be the instance with the higher version.

2) If the MinorVersion properties of the two instances are equal, go to step 3.

   Otherwise, the instance with the higher value of the MinorVersion property shall be the instance with the higher version.

3) If the RevisionNumber properties of the two instances are equal, go to step 4.

   Otherwise, the instance with the higher value of the RevisionNumber property shall be the instance with the higher version.

4) If the BuildNumber properties of the two instances are equal, the two instances shall have equal value.

   Otherwise, the instance with the higher value of the BuildNumber property shall be the instance with the higher version.

# 8 Methods

This section details the requirements for supporting intrinsic operations for the CIM elements defined by this profile. The *Software Inventory Profile* does not define any extrinsic methods.

## 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 the statement “All operations in the default list in section 8.1 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 than that described by DSP0200 version 1.2.

The default list of operations is as follows:

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

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

## 8.2 CIM_SoftwareIdentity

All operations in the default list in section 8.1 are supported as described by DSP0200 version 1.2.

## 8.3 CIM_InstalledSoftwareIdentity

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_InstalledSoftwareIdentity**

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

## 8.4 CIM_ElementSoftwareIdentity

Table 4 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 4 – Operations: CIM_ElementSoftwareIdentity**

| Operation | Requirement | Messages |
|---|---|---|
| ModifyInstance | Optional. See section 8.4.1. | None |
| Associators | Unspecified | None |
| AssociatorNames | Unspecified | None |
| References | Unspecified | None |
| ReferenceNames | Unspecified | None |

### 8.4.1 CIM_ElementSoftwareIdentity – ModifyInstance

The following rules shall dictate the behavior of the ModifyInstance operation:

- When the ModifyInstance operation is used to set the ElementSoftwareStatus property to contain the value 3 (Next):
    1) find all the other instances of CIM_ElementSoftwareIdentity that
        a) reference the same instance of CIM_ManagedElement as the target instance of CIM_ElementSoftwareIdentity and
        b) reference an instance of CIM_SoftwareIdentity that belongs to the same Software Family as the instance of CIM_SoftwareIdentity that is referenced by the target instance of CIM_ElementSoftwareIdentity.
    2) For each of the CIM_ElementSoftwareIdentity instances found, remove the value 3 (Next) from the ElementSoftwareStatus property if present.
- The implementation shall not allow the ModifyInstance operation to add the value 2 (Current) to and remove the value 2 (Current) from the ElementSoftwareStatus property.
- When the ModifyInstance operation is used to set the ElementSoftwareStatus property to contain the value 4 (FallBack):

1) Find all the other instances of CIM_ElementSoftwareIdentity that

   a) reference the same instance of CIM_ManagedElement as the target instance of CIM_ElementSoftwareIdentity and

   b) reference an instance of CIM_SoftwareIdentity that belongs to the same Software Family as the instance of CIM_SoftwareIdentity that is referenced by the target instance of CIM_ElementSoftwareIdentity.

2) For each of the CIM_ElementSoftwareIdentity instances found, remove the value 4 (FallBack) from the ElementSoftwareStatus property if present.

- The implementation shall not allow the ModifyInstance operation to add or remove the value 5 (Default) from the ElementSoftwareStatus property.
- The implementation shall not allow the ModifyInstance operation to add or remove the value 6 (Installed) from the ElementSoftwareStatus property.
- When the ModifyInstance operation is used to set the ElementSoftwareStatus property to contain the value 7 (SingleUse):

1) Find all the other instances of CIM_ElementSoftwareIdentity that

   a) reference the same instance of CIM_ManagedElement as the target instance of CIM_ElementSoftwareIdentity and

   b) reference an instance of CIM_SoftwareIdentity that belongs to the same Software Family as the instance of CIM_SoftwareIdentity that is referenced by the target instance of CIM_ElementSoftwareIdentity.

2) For each of the CIM_ElementSoftwareIdentity instances found, remove the value 7 (SingleUse) from the ElementSoftwareStatus property if present.

- The implementation shall not allow the ModifyInstance operation to remove the value 8 (Available) from the ElementSoftwareStatus property. The implementation shall allow adding the value 8 (Available) to the ElementSoftwareStatus property only if the associated Software Identity is associated with the CIM_SystemSpecificCollection that has the ElementName property equal to “Available Software” through an instance of CIM_MemberOfCollection.
- The implementation shall not allow the ModifyInstance operation to add or remove the value 9 (Supports) from the ElementSoftwareStatus property.

## 8.5 CIM_SystemSpecificCollection

All operations in the default list in section 8.1 are supported as described by DSP0200 version 1.2.

## 8.6 CIM_HostedCollection

Table 5 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 5 – Operations: CIM_HostedCollection**

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

## 8.7 CIM_MemberOfCollection

Table 6 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 6 – Operations: CIM_MemberOfCollection**

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

## 8.8 CIM_SoftwareIdentityResource

All operations in the default list in section 8.1 are supported as described by DSP0200 version 1.2.

## 8.9 CIM_SAPAvailableForElement

Table 7 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 7 – Operations: CIM_SAPAvailableForElement**

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

## 8.10 CIM_HostedAccessPoint

Table 8 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 8 – Operations: CIM_HostedAccessPoint**

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

## 8.11 CIM_OrderedComponent

Table 9 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 9 – Operations: CIM_OrderedComponent**

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

### 8.12 CIM_OrderedDependency

Table 10 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

**Table 10 – Operations: CIM_OrderedDependency**

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

# 9 Use Cases

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

## 9.1 Object Diagrams

This section contains object diagrams for the *Software Inventory Profile.* For simplicity, the prefix *CIM_* has been removed from the names of the classes in the diagrams.

### 9.1.1 Registered Profile

Figure 2 represents a possible instantiation of the *Software Inventory Profile*. In this instantiation, the Central Instance, swid1, has an InstalledSoftwareIdentity association to the Scoping Instance, system1. Profile registration information is represented with the profile1 instance. Following the CIM_ElementConformsToProfile association from the Central Instance to profile1, the client can retrieve information such as the version of the current *Software Inventory Profile* implementation.

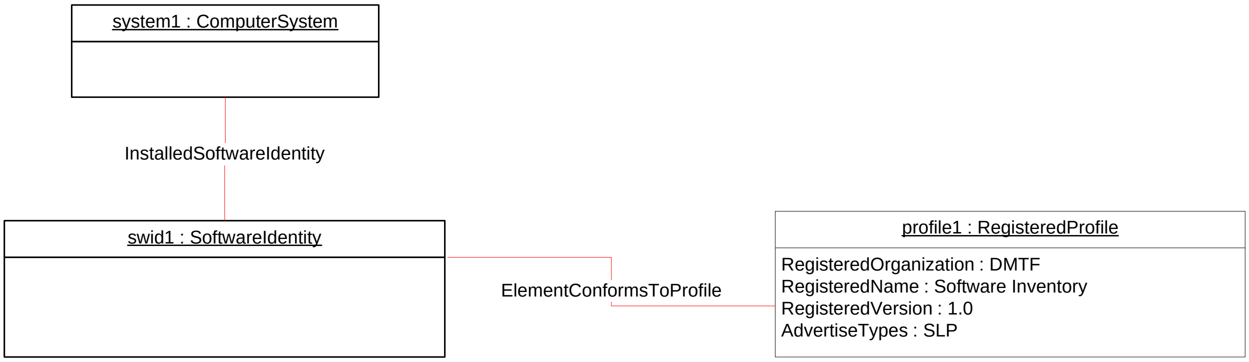

system1 : ComputerSystem
InstalledSoftwareIdentity
swid1 : SoftwareIdentity
ElementConformsToProfile
profile1 : RegisteredProfile
RegisteredOrganization : DMTF
RegisteredName : Software Inventory
RegisteredVersion : 1.0
AdvertiseTypes : SLP

**Figure 2 – Registered Profile**

### 9.1.2 Representing Installed Firmware

Figure 3 represents a possible instantiation of the *Software Inventory Profile*. In the object diagram, Software Identity swid1 is shown as installed on the SCSI PCI Controller, scsictrlr1, and currently running on it. The ElementSoftwareStatus property on the ElementSoftwareIdentity association instance between swid1 and scsictrl1 does not have the value 3 (Next) because it is not the firmware that will run after the next reboot of the system.

Software Identity swid2 is shown as installed on the Network PCI Controller, pcictrlr1, and currently running on it. swid2 would also run on the next reset or reboot of pcictlr1. The object diagram does not show the CIM_SystemDevice association between system1 and scsictrl1, and system1 and pcictrl1, but both scsictrl1 and pcictrl1 are scoped to system1 and so the CIM_InstalledSoftwareIdentity association is shown between system1 and swid1, and system1 and swid2.

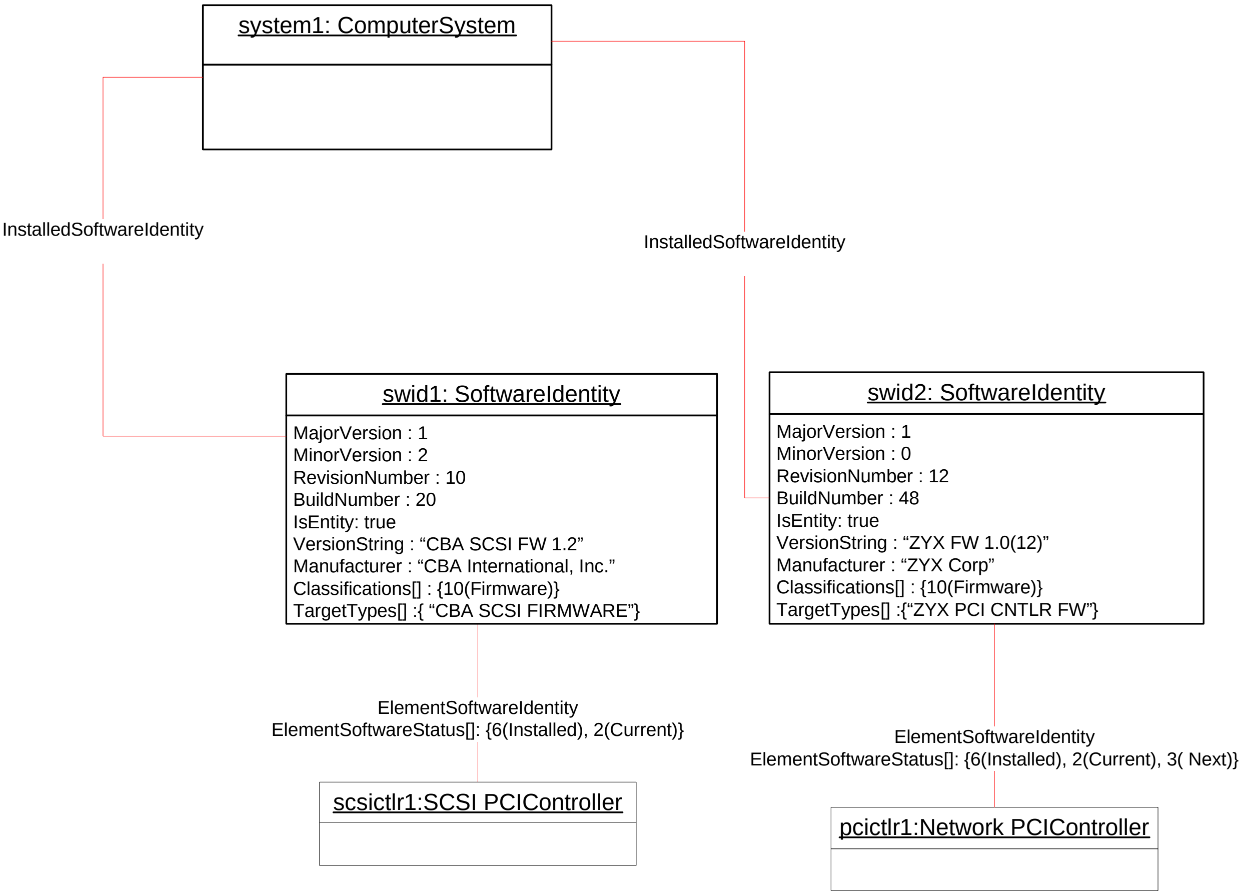

system1: ComputerSystem
InstalledSoftwareIdentity
InstalledSoftwareIdentity
swid1: SoftwareIdentity
MajorVersion : 1
MinorVersion : 2
RevisionNumber : 10
BuildNumber : 20
IsEntity: true
VersionString : “CBA SCSI FW 1.2”
Manufacturer : “CBA International, Inc.”
Classifications[] : {10(Firmware)}
TargetTypes[] :{ “CBA SCSI FIRMWARE”}
swid2: SoftwareIdentity
MajorVersion : 1
MinorVersion : 0
RevisionNumber : 12
BuildNumber : 48
IsEntity: true
VersionString : “ZYX FW 1.0(12)”
Manufacturer : “ZYX Corp”
Classifications[] : {10(Firmware)}
TargetTypes[] :{“ZYX PCI CNTLR FW”}
ElementSoftwareIdentity
ElementSoftwareStatus[]: {6(Installed), 2(Current)}
ElementSoftwareIdentity
ElementSoftwareStatus[]: {6(Installed), 2(Current), 3( Next)}
scsictlr1:SCSI PCIController
pcictlr1:Network PCIController

**Figure 3 – Object Diagram Showing Installed Software**

### 9.1.3 Representing an Installed Driver

Figure 4 represents a possible instantiation of the *Software Inventory Profile*. It shows how to model an installed driver. In this instantiation, the driver, dr1, is applicable to the NIC, nic1. The ElementSoftwareStatus value “Supports” indicates that dr1 is applicable to nic1. The driver is installed in the OS, os1, and is the driver for nic1 that is currently running in os1.

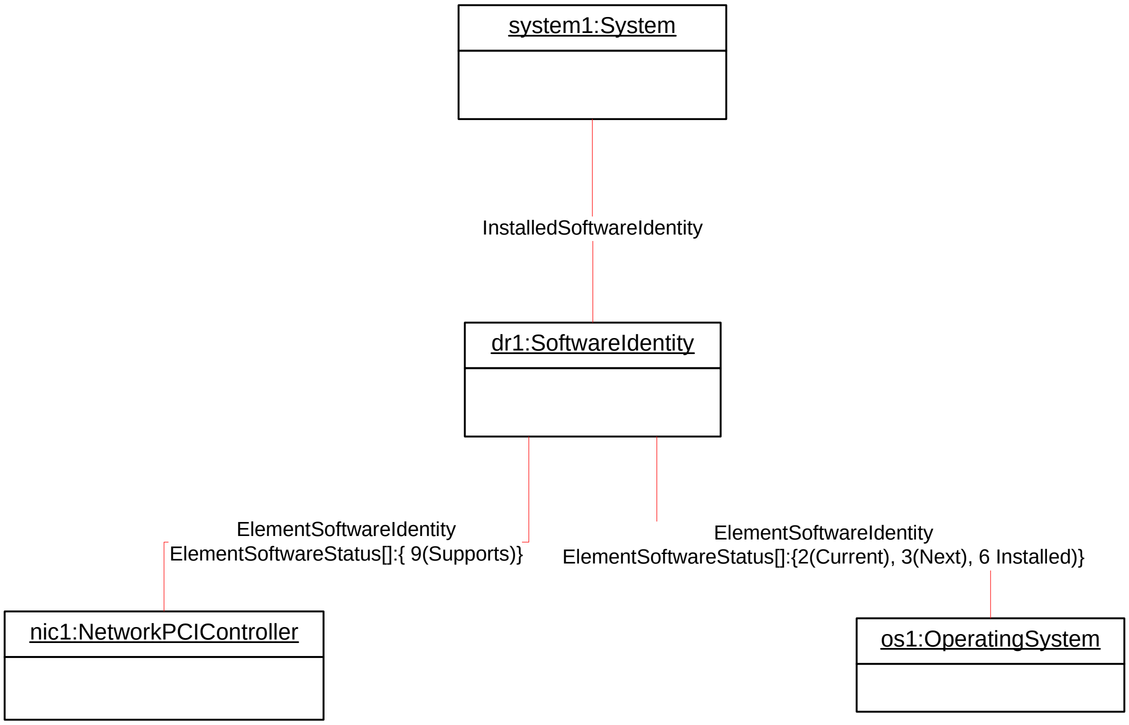

system1:System
InstalledSoftwareIdentity
dr1:SoftwareIdentity
ElementSoftwareIdentity
ElementSoftwareStatus[]:{ 9(Supports)}
ElementSoftwareIdentity
ElementSoftwareStatus[]:{2(Current), 3(Next), 6 Installed)}
nic1:NetworkPCIController
os1:OperatingSystem

**Figure 4 – Object Diagram Showing an Installed Driver**

### 9.1.4 Representing BIOS Installed on a System

Figure 5 represents a possible instantiation of the *Software Inventory Profile*. Both bios1 and bios2 are associated with system1 through an instance of InstalledSoftwareIdentity because both of them are installed on a component of the system, which happens to be the system itself.

bios1 is for the system, system1, and so the CIM_ElementSoftwareIdentity association is used to associate them with the ElementSoftwareStatus property having the values 2 (Current), 3 (Next), and 6 (Installed).

bios2 is the backup for bios1 and is also for system, system1, and so the CIM_ElementSoftwareIdentity association is used to associate them with the ElementSoftwareStatus property having the values 4 (FallBack) and 6 (Installed).

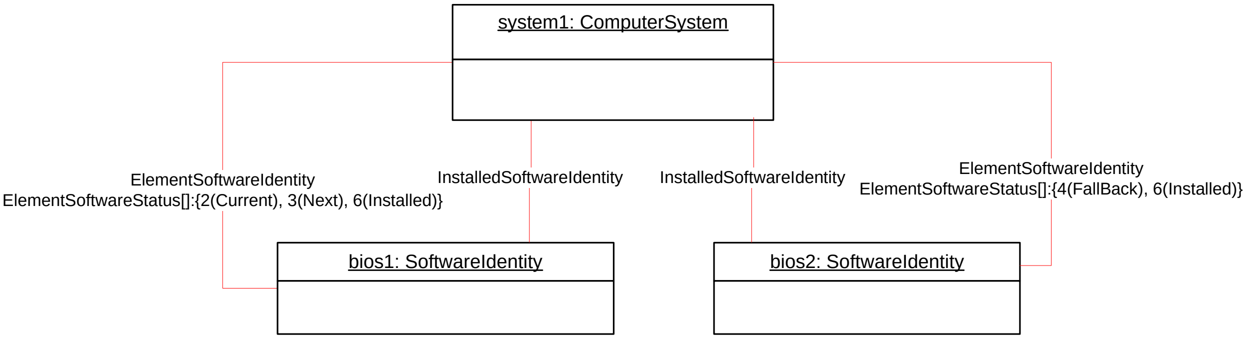

system1: ComputerSystem
ElementSoftwareIdentity
ElementSoftwareStatus[]:{2(Current), 3(Next), 6(Installed)}
InstalledSoftwareIdentity
InstalledSoftwareIdentity
ElementSoftwareIdentity
ElementSoftwareStatus[]:{4(FallBack), 6(Installed)}
bios1: SoftwareIdentity
bios2: SoftwareIdentity

**Figure 5 – Object Diagram Showing Installed BIOS**

### 9.1.5 Representing Installed Software without Any Association to the Managed Element

Figure 6 represents a possible instantiation of the *Software Inventory Profile*. The firmware represented by swid1 is installed on some Managed Element in the scope of system1 but the Managed Element is not modeled by the instrumentation and since the CIM_ElementSoftwareIdentity association is not instantiated between system1 and swid1, the relationship the Managed Element and swid1 is not known.

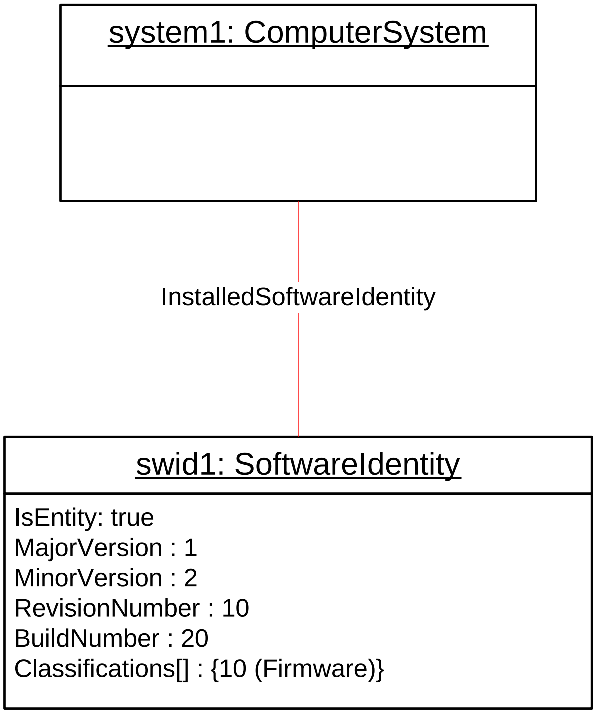


**Figure 6 – Object Diagram Showing Installed Software**

### 9.1.6 Representing More Than One Executable Software Identity on a Managed Element

Figure 7 represents a possible instantiation of the *Software Inventory Profile*. In the object diagram, Software Identity swid2 is shown as installed on the SCSI PCI Controller, scsictrlr1, and is currently running on it. The ElementSoftwareStatus property on the CIM_ElementSoftwareIdentity instance that associates swid2 and scsictrl1 has the values 2 (Current) and 6 (Installed).

Software Identity swid3 is the manufacturer shipped version and is installed on scsictrlr1 but is not currently running. The CIM_ElementSoftwareIdentity instance that associates swid3 and scsictrl1 conveys this relationship by the ElementSoftwareStatus property having the values 5 (Default) and 6 (Installed).

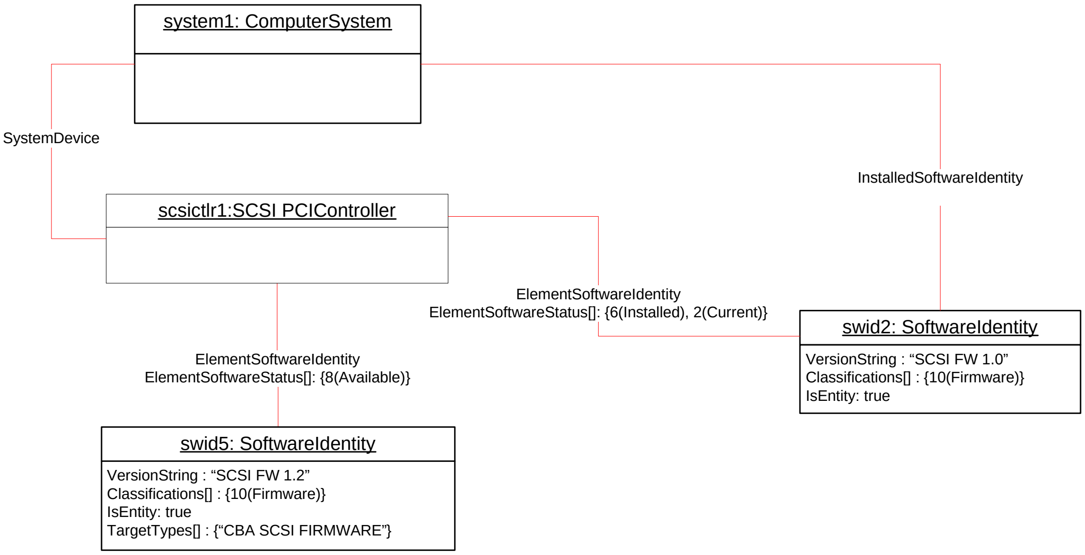

system1: ComputerSystem
SystemDevice
InstalledSoftwareIdentity
scsictlr1:SCSI PCIController
ElementSoftwareIdentity
ElementSoftwareStatus[]: {6(Installed), 2(Current)}
swid2: SoftwareIdentity
VersionString : "SCSI FW 1.0"
Classifications[] : {10(Firmware)}
IsEntity: true
ElementSoftwareIdentity
ElementSoftwareStatus[]: {8(Available)}
swid5: SoftwareIdentity
VersionString : "SCSI FW 1.2"
Classifications[] : {10(Firmware)}
IsEntity: true
TargetTypes[] : {"CBA SCSI FIRMWARE"}

**Figure 7 – Object Diagram Showing Multiple Installed Software on a Managed Element**

### 9.1.7 Representing Available and Installed Firmware without Managed Element

Figure 8 represents a possible instantiation of the *Software Inventory Profile*. The object diagram is an alternative instantiation of Figure 7 where the SCSI PCI Controller, scsictrlr1, is not instantiated. Thus swid2 and swid5 are associated through the CIM_ElementSoftwareIdentity associations to the Scoping Instance, system1.

The ElementSoftwareStatus property on these associations still represents the relationship between the SCSI PCI Controller and swid2 and swid5.

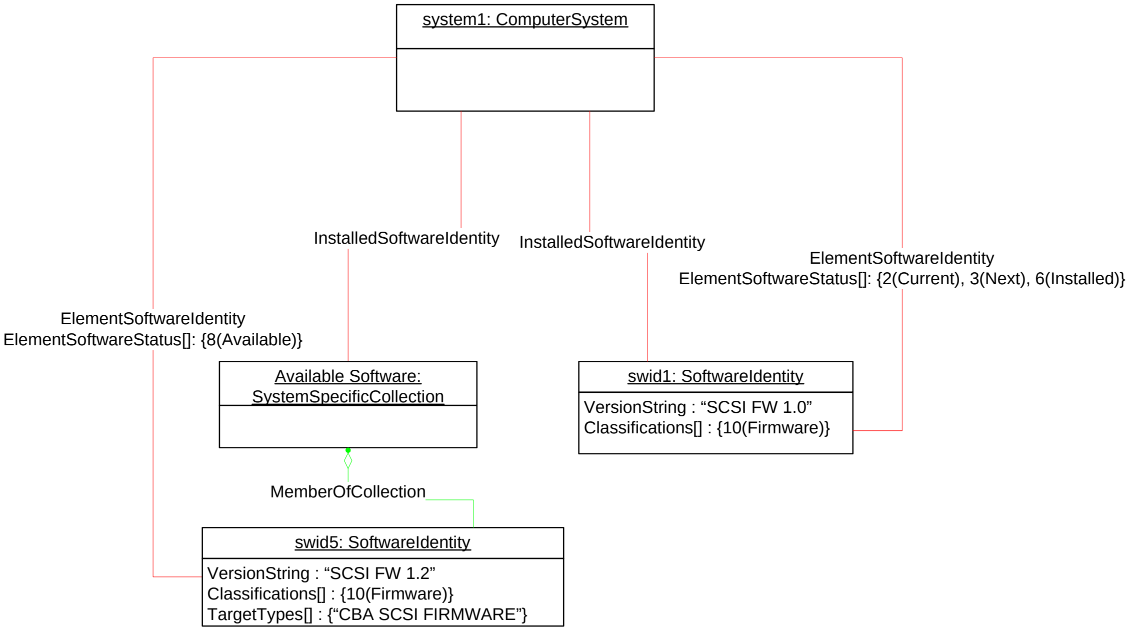

system1: ComputerSystem
InstalledSoftwareIdentity
InstalledSoftwareIdentity
ElementSoftwareIdentity
ElementSoftwareStatus[]: {2(Current), 3(Next), 6(Installed)}
ElementSoftwareIdentity
ElementSoftwareStatus[]: {8(Available)}
Available Software:
SystemSpecificCollection
swid1: SoftwareIdentity
VersionString : "SCSI FW 1.0"
Classifications[] : {10(Firmware)}
MemberOfCollection
swid5: SoftwareIdentity
VersionString : "SCSI FW 1.2"
Classifications[] : {10(Firmware)}
TargetTypes[] : {"CBA SCSI FIRMWARE"}

**Figure 8 – Object Diagram with No Instantiation of Managed Element**

### 9.1.8 Representing Available Firmware

Figure 9 represents a possible instantiation of the *Software Inventory Profile*. In the object diagram, Software Identity swid2 is shown as available for installation on the Network PCI Controller, pcictrlr1, using the CIM_ElementSoftwareIdentity association. Software Identity swid1 is an Available Software but the compatible Managed Element is not modeled and no CIM_ElementSoftwareIdentity instance references swid1. pcictrl1 is scoped to system1, but the object diagram does not show the CIM_SystemDevice association between system1 and pcictrl1, and so the CIM_MemberOfCollection association is shown between an "Available Software" collection and swid2.

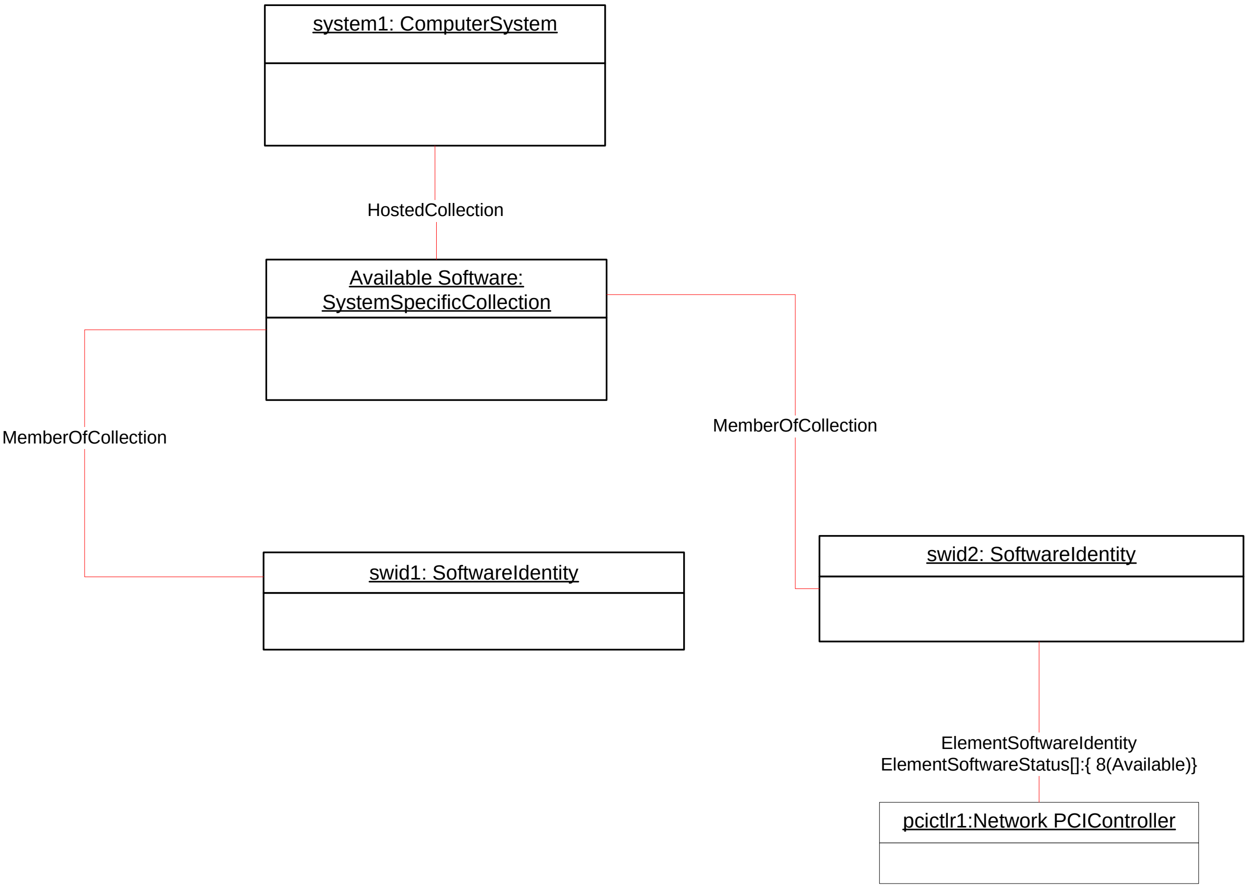
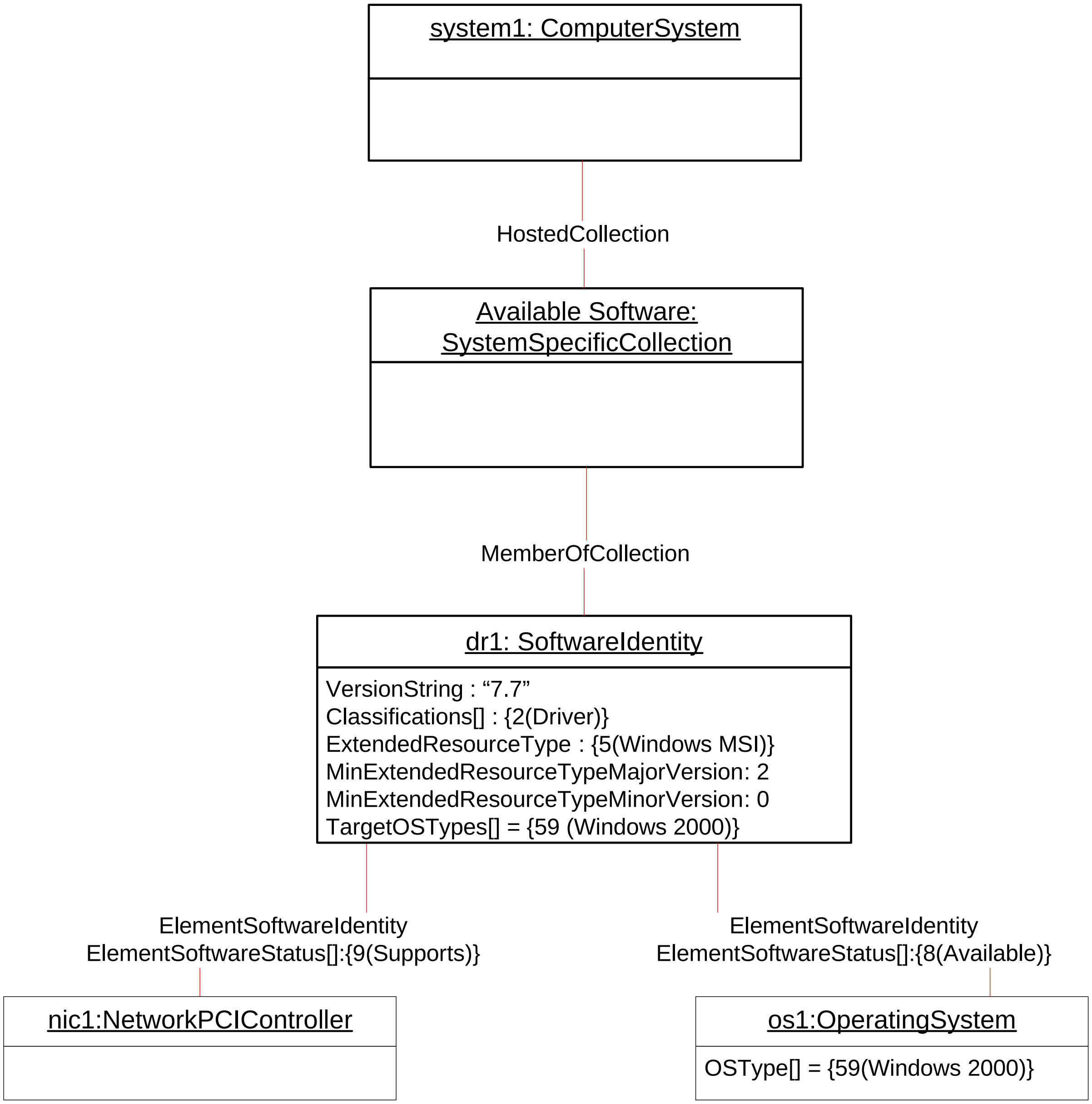


**Figure 9 – Object Diagram Showing Available Firmware**

### 9.1.9 Representing an Available Driver and Its Relationship to the Operating System

Figure 10 represents a possible instantiation of the *Software Inventory Profile*: an available driver. In this instantiation, the driver, dr1, is applicable to the NIC, nic1. The ElementSoftwareStatus property of the CIM_ElementSoftwareIdentity association instance between dr1 and nic1 has the value 9 (Supports), indicating that dr1 is applicable to nic1. The object diagram also represents the driver's relationship to the operating system, os1, with the ElementSoftwareIdentity association instance having the ElementSoftwareStatus property with the value 8 (Available), indicating that dr1 is applicable to os1 and is available for installation. The relationship between system1 and os1 is not shown.


system1: ComputerSystem
HostedCollection
Available Software:
SystemSpecificCollection
MemberOfCollection
dr1: SoftwareIdentity
VersionString : “7.7”
Classifications[] : {2(Driver)}
ExtendedResourceType : {5(Windows MSI)}
MinExtendedResourceTypeMajorVersion: 2
MinExtendedResourceTypeMinorVersion: 0
TargetOSTypes[] = {59 (Windows 2000)}
ElementSoftwareIdentity
ElementSoftwareStatus[]:{9(Supports)}
ElementSoftwareIdentity
ElementSoftwareStatus[]:{8(Available)}
nic1:NetworkPCIController
os1:OperatingSystem
OSType[] = {59(Windows 2000)}

730

731 **Figure 10 – Object Diagram Showing an Available Driver**

### 9.1.10 Representing Available Software and Its Location Information

Figure 11 represents a possible instantiation of the *Software Inventory Profile*: an Available Software and its location information. In this instantiation, the firmware, fw1, is available to the system and its location information is modeled by swidres1.

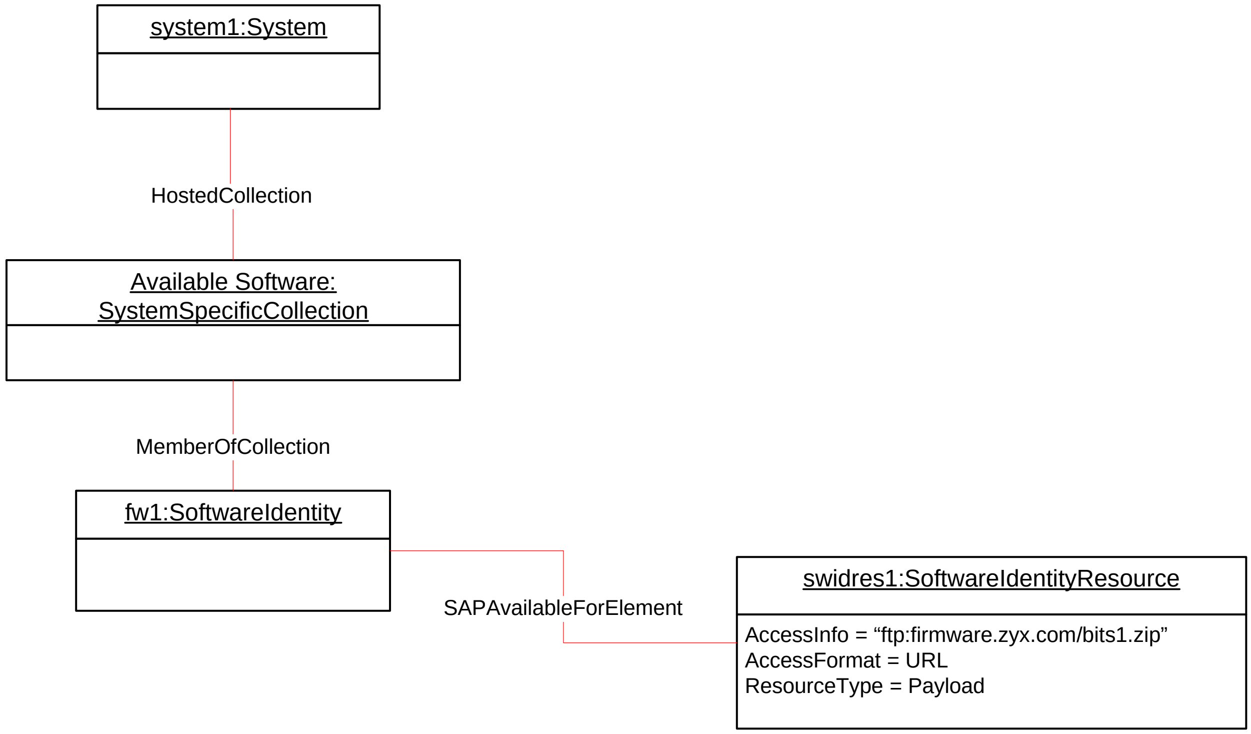


**Figure 11 – Object Diagram Showing a Firmware Image and Its Location**

**EXPERIMENTAL**

### 9.1.11 Representing a Software Bundle

Figure 12 represents a possible instantiation of the *Software Inventory Profile*: a Software Bundle. In the diagram, the Software Bundle, bundle1, consists of two Software Identities:

- dr1 with the Assigned sequence of 1, indicating that dr1 will be the first to be installed while installing the bundle
- bios1 with the Assigned sequence of 2, indicating that bios1 will be the second to be installed while installing the bundle

After bundle1 has been installed, instrumentation will create associations relating to dr1 as shown in Figure 4 and associations relating to bios1 as shown in Figure 5.

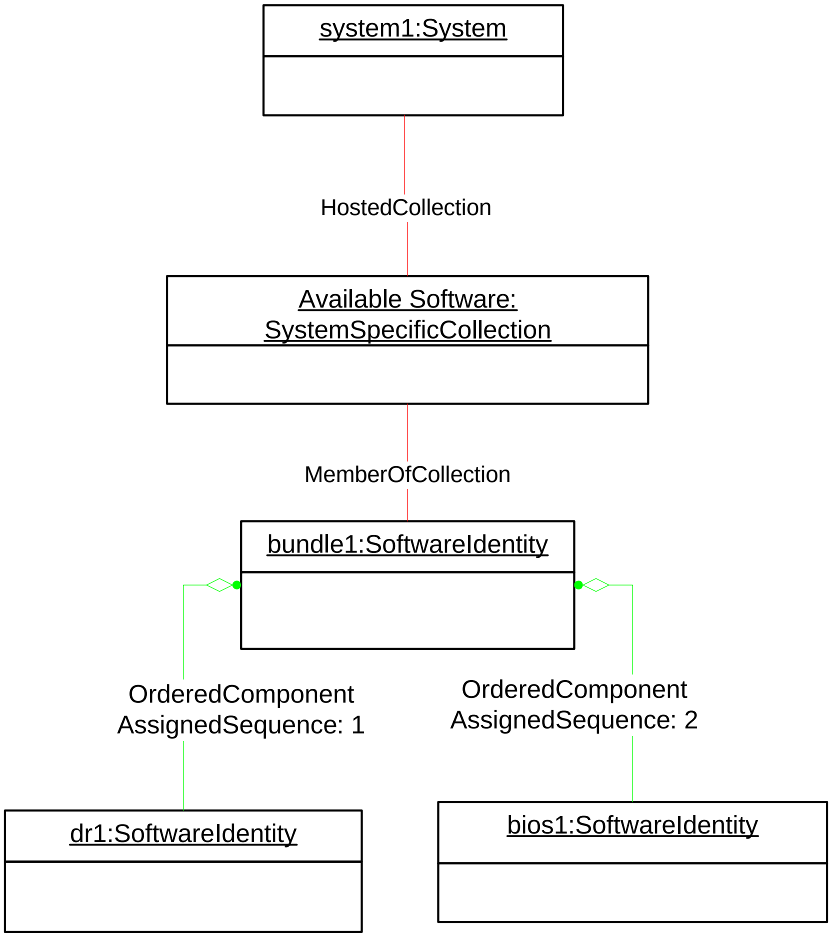


**Figure 12 – Object Diagram Showing a Software Bundle**

**EXPERIMENTAL**

**EXPERIMENTAL**

### 9.1.12 Representing Software That Is Part of a Software Bundle and Available

Figure 13 represents a possible instantiation of the *Software Inventory Profile*. In the diagram, the Software Bundle, bundle2, consists of two Software Identities, fw2 and dr2, both of which are members of the “Available Software” collection. So, fw2 and dr2 could be installed either individually or as a part of installing bundle2.

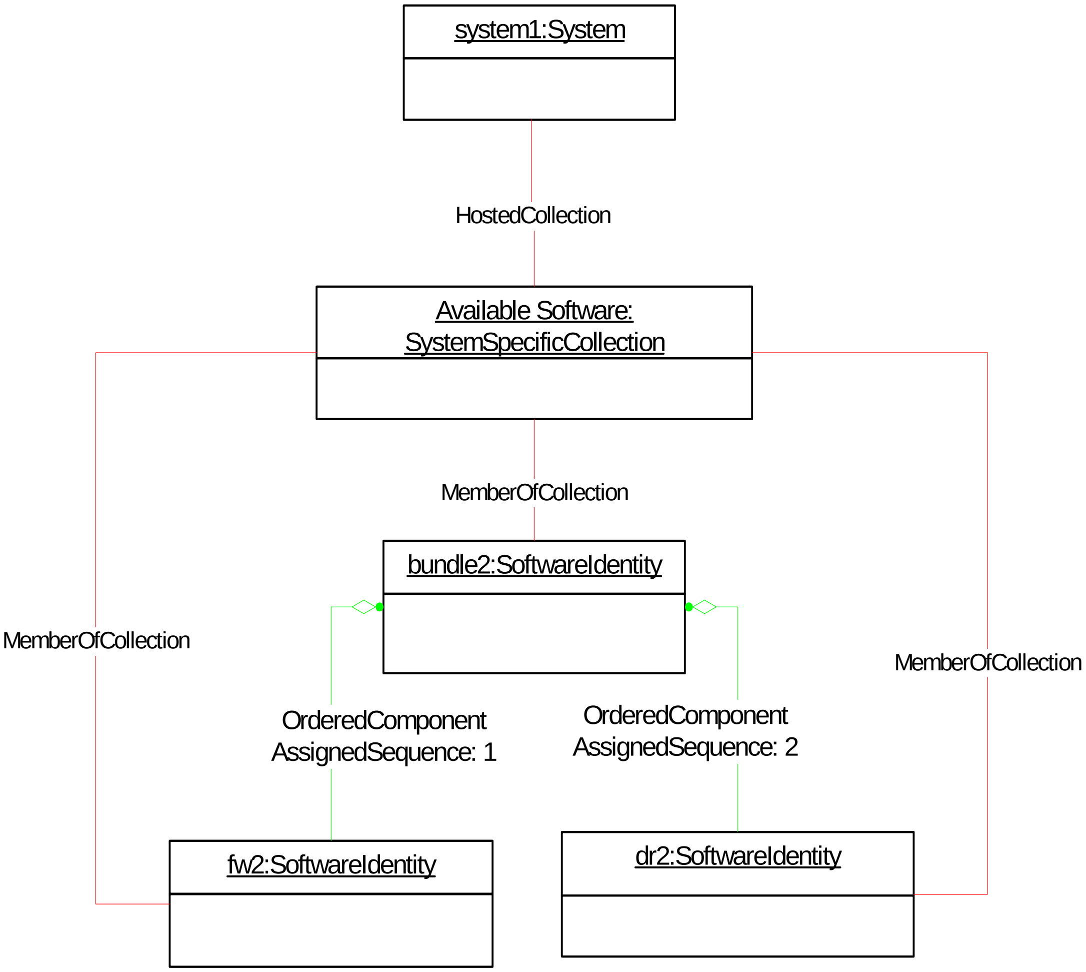


**Figure 13 – Object Diagram Showing Available Software That Is Part of a Software Bundle**

**EXPERIMENTAL**

**EXPERIMENTAL**

### 9.1.13 Representing Installed and Available Software

Figure 14 represents a possible instantiation of the *Software Inventory Profile*. In the object diagram, Software Identity swid2 is shown as installed on the SCSI PCI Controller, scsictrlr1, and is currently running on it. The ElementSoftwareStatus property on the CIM_ElementSoftwareIdentity instance associating swid2 and scsictrl1 has the values 2 (Current) and 6 (Installed).

Software Identity swid5 is shown as Available Software for scsictrlr1, and so the ElementSoftwareStatus property on the CIM_ElementSoftwareIdentity instance that associates swid5 and scsictrl1 has the value 8 (Available).

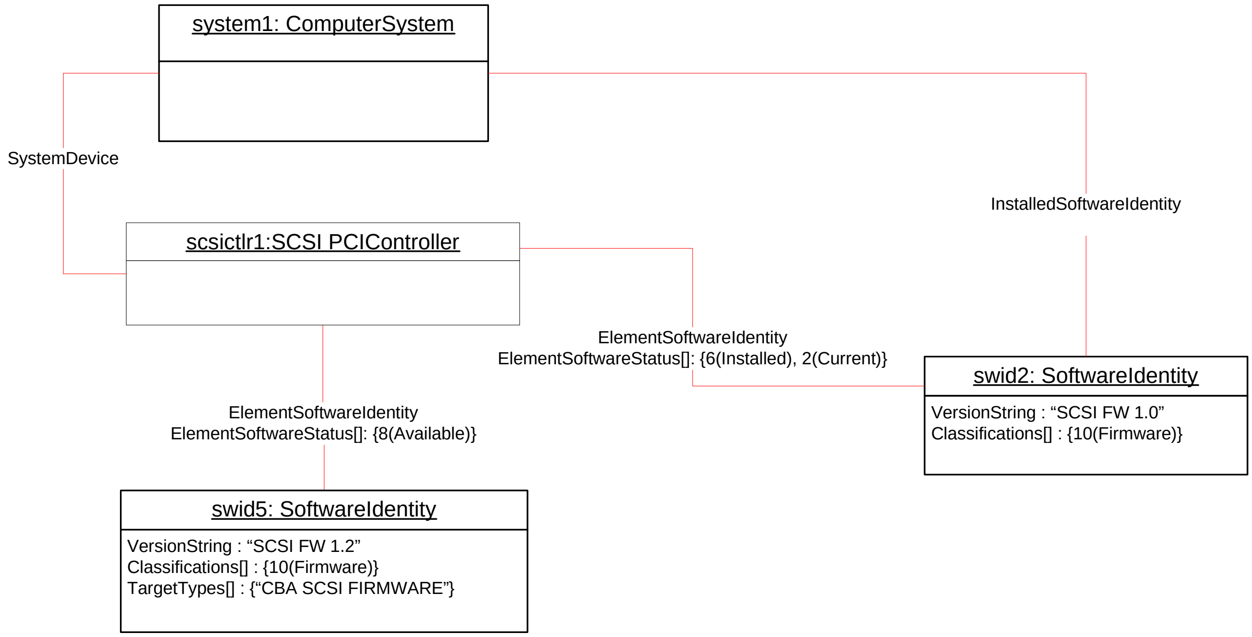

system1: ComputerSystem
SystemDevice
InstalledSoftwareIdentity
scsictlr1:SCSI PCIController
ElementSoftwareIdentity
ElementSoftwareStatus[]: {6(Installed), 2(Current)}
swid2: SoftwareIdentity
VersionString : "SCSI FW 1.0"
Classifications[] : {10(Firmware)}
ElementSoftwareIdentity
ElementSoftwareStatus[]: {8(Available)}
swid5: SoftwareIdentity
VersionString : "SCSI FW 1.2"
Classifications[] : {10(Firmware)}
TargetTypes[] : {"CBA SCSI FIRMWARE"}

**Figure 14 – Object Diagram Showing Installed and Available Software**

**EXPERIMENTAL**

## 9.2 Find All the Software Installed on All the Managed Elements within the Scope of a Managed System

For the instance of CIM_System that represents the given managed system, select all the instances of CIM_SoftwareIdentity that are associated through instances of CIM_InstalledSoftwareIdentity. The resulting instances represent the software installed on all the Managed Elements in the scope of the managed system.

## 9.3 Find All the Software Installed on a Managed Element

For the given instance of CIM_ManagedElement, select the instance of CIM_SoftwareIdentity that is associated through an instance of CIM_ElementSoftwareIdentity such that the ElementSoftwareStatus property contains the value 6 (Installed).

## 9.4 Find All the Software That Is Compatible with a Managed Element but Has Not Been Installed

For the given instance of CIM_ManagedElement, using the CIM_ElementSoftwareIdentity association, select the associated instances of CIM_SoftwareIdentity that are not associated to the scoping CIM_System or CIM_ComputerSystem instance through the CIM_InstalledSoftwareIdentity association. If the given instance of CIM_ManagedElement does not have any associating CIM_ElementSoftwareIdentity instances referencing it, the compatible software will not be determinable and no instances of CIM_SoftwareIdentity will be returned.

## 9.5 Find All the Software That Is Available for Installation on Any Managed Element within the Scope of a Managed System

For the instance of CIM_ComputerSystem that represents the given managed system, select the instance of CIM_SystemSpecificCollection with ElementName value of “Available Software” that is associated through and instance of CIM_HostedCollection. Select all the instances of CIM_SoftwareIdentity that are associated through an instance of CIM_MemberOfCollection.

## 9.6 For a Given NIC, Find the Driver That Is Running in the Operating System

The client can find the driver that is currently running by using the following steps:

1) For the instance of CIM_ManagedElement that represents the NIC, select the instances of the CIM_ElementSoftwareIdentity association with the ElementSoftwareStatus property containing the value 9 (Supports).

2) Select the instances of CIM_SoftwareIdentity that the instances of CIM_ElementSoftwareIdentity from step 1 reference.

3) From the given instance of CIM_ManagedElement that represents the NIC, select the instance of CIM_ComputerSystem that is associated through an instance of CIM_SystemDevice.

4) From the CIM_ComputerSystem instance from step 3, select the instance of CIM_OperatingSystem that is associated through an instance of CIM_RunningOS.

5) Select the instances of CIM_ElementSoftwareIdentity that reference the instance of CIM_OperatingSystem from step 4 and contain the value 2 (Current) in the ElementSoftwareStatus property.

6) Select the instance of CIM_SoftwareIdentity that is referenced by at least one instance of CIM_ElementSoftwareIdentity from step 2 and at least one instance of CIM_ElementSoftwareIdentity from step 5.

## 9.7 Set a Particular Software Image on a Hardware Managed Element to Run After the Next Reset or Reboot

The client can set a particular software image on a hardware managed element to run after the next reset or reboot by using the following steps:

1) Select the CIM_ElementSoftwareIdentity association instance that associates the Managed Element instance that represents the device with the Software Identity instance that represents the software image.

2) Set the value of the ElementSoftwareStatus property on the ElementSoftwareIdentity association to 3 (Next).

## 9.8 Set a Particular Software Image on a Hardware Managed Element to Run After the Next Reset or Reboot But Not After a Subsequent Reset or Reboot

The client can set a particular software image on a hardware managed element to run after the next reset or reboot but not after a subsequent reset or reboot by using the following steps:

1) Select the CIM_ElementSoftwareIdentity association instance that associates the Managed Element instance that represents the device with the Software Identity instance that represents the software image.

2) Set the value of the ElementSoftwareStatus property on the ElementSoftwareIdentity association to 7 (SingleUse).

## 9.9 Find and Set a Driver to Run After the Next Reset or Reboot for a NIC

A client can set a driver to run on the next reset or reboot by using the following steps:

1) For the instance of CIM_ManagedElement that represents the NIC, select the instances of CIM_ElementSoftwareIdentity association with the ElementSoftwareStatus property containing the value 9 (Supports).

2) Select the instance of CIM_SoftwareIdentity that the instances of CIM_ElementSoftwareIdentity from step 1 reference.

3) Identify the CIM_SoftwareIdentity instance that corresponds to the driver.

4) From the given instance of CIM_ManagedElement that represents the NIC, select the instance of CIM_ComputerSystem that is associated through an instance of CIM_SystemDevice.

5) From the CIM_ComputerSystem instance from step 4, select the instance of CIM_OperatingSystem that is associated through an instance of CIM_RunningOS.

6) Select the instances of CIM_ElementSoftwareIdentity that reference the instance of CIM_OperatingSystem from step 5 and contain the value 6 (Installed) in the ElementSoftwareStatus property.

7) Select the instance of CIM_ElementSoftwareIdentity that associates the instance of CIM_ManagedElement and the instance of CIM_SoftwareIdentity from step 3. Set the value of the ElementSoftwareStatus property of this instance to 3 (Next).

## 9.10 Find the Most Recent Firmware Available for a NIC

A client can find the most recent firmware available for a NIC by using the following steps:

1) For the given instance of CIM_ManagedElement that represents the NIC, select the instances of CIM_SoftwareIdentity that are associated through instances of CIM_ElementSoftwareIdentity with the ElementSoftwareStatus property containing the value 8 (Available) with the Classifications[] property on the CIM_SoftwareIdentity instance containing the value 10 (Firmware).

2) From the instances returned, select the instance of CIM_SoftwareIdentity with the highest version. (See section 7.10 for the version comparison algorithm.)

## 9.11 Find the Most Recent Firmware Installed on a NIC

A client can find the most recent firmware installed on a NIC by using the following steps:

1) For the given instance of CIM_ManagedElement that represents the NIC, select the instances of CIM_SoftwareIdentity that are associated through instances of CIM_ElementSoftwareIdentity with the ElementSoftwareStatus property containing the value 6 (Installed) with the Classifications[] property on the CIM_SoftwareIdentity instance containing the value 10 (Firmware).

2) From the instances returned, select the instance of CIM_SoftwareIdentity with the highest version. (See section 7.10 for the version comparison algorithm.)

## 9.12 Find the Software Families of Which a Software Identity Is a Member

For the given instance of CIM_SoftwareIdentity, select all the values in the IdentityInfoValue[] property array that have a value at the corresponding index in the IdentityInfoType[] property array equal to “CIM:SoftwareFamily”. Each of the selected values represents a Software Family of which the Software Identity is a member.

## 9.13 Determine Whether a Dependency of a Software Identity Is Satisfied

Given an instance of CIM_SoftwareIdentity that represents an Installation Dependency for a Software Identity, a client can determine if the dependency is resolved as follows:

1) From the Scoping Instance, select all the instances of CIM_SoftwareIdentity that are associated through instances of CIM_InstalledSoftwareIdentity.

2) For each Software Identity from step 1, determine all the Software Families to which it belongs by using the algorithm in section 9.12.

3) For the instance of CIM_SoftwareIdentity that represents the dependency, determine the Software Families by using the algorithm in section 9.12.

4) Select the instance of CIM_SoftwareIdentity from step 1 such that at least one Software Family to which it belongs (from step 2) is equal to at least one Software Family to which the dependency belongs (from step 3).

The dependency is satisfied if the version of the selected Software Identity is greater than or equal to the version of the dependency represented by an instance of CIM_SoftwareIdentity. (See section 7.10 for the version comparison algorithm.)

# 10 CIM Elements

Table 11 shows the instances of CIM Elements for this profile. Instances of the CIM Elements shall be implemented as described in Table 11. Sections 7 (“Implementation”) and 8 (“Methods”) may impose additional requirements on these elements.

**Table 11 – CIM Elements: Software Inventory Profile**

| Element Name | Requirement | Description |
|---|---|---|
| **Classes** | | |
| CIM_SoftwareIdentity | Mandatory | See sections 7.2, 7.6, and 10.1. |
| CIM_InstalledSoftwareIdentity | Conditional | See sections 7.2.1 and 10.2. |
| CIM_ElementSoftwareIdentity | Optional | See sections 7.4 and 10.3. |
| CIM_SystemSpecificCollection | Optional | See sections 7.6.1 and 10.4. |
| CIM_HostedCollection | Conditional | See sections 7.6.2 and 10.5. |
| CIM_MemberOfCollection | Conditional | See sections 7.6.3 and 10.6. |
| CIM_SoftwareIdentityResource | Optional | See sections 7.6.4.1 and 10.7.<br>EXPERIMENTAL |
| CIM_SAPAvailableForElement | Conditional | See sections 7.6.4.2 and 10.8. |
| CIM_HostedAccessPoint | Optional | See sections 7.6.4.3 and 10.9. |
| CIM_OrderedComponent | Optional | See sections 7.7 and 10.10.<br>EXPERIMENTAL |

| Element Name | Requirement | Description |
|---|---|---|
| CIM_OrderedDependency | Optional | See sections 7.9.1 and 10.11.<br>EXPERIMENTAL |
| CIM_RegisteredProfile | Mandatory | See section 10.12. |
| **Indications** | | |
| None defined in this profile | | |

## 10.1 CIM_SoftwareIdentity

CIM_SoftwareIdentity is used to represent either Installed Software or Available Software. Table 12 contains the requirements for elements of this class.

**Table 12 – Class: CIM_SoftwareIdentity**

| Elements | Requirement | Notes |
|---|---|---|
| InstanceID | Mandatory | **Key** |
| IsEntity | Mandatory | See sections 7.2, 7.6, and 7.9. |
| VersionString | Optional | |
| MajorVersion | Conditional | See section 7.3. |
| MinorVersion | Conditional | See section 7.3. |
| RevisionNumber | Conditional | See section 7.3. |
| BuildNumber | Conditional | See section 7.3. |
| TargetOSTypes[] | Optional | See section 7.6.5. |
| TargetOperatingSystems[] | Optional | See section 7.6.5. |
| IdentityInfoType[] | Optional | See section 7.8.2. |
| IdentityInfoValue[] | Optional | See section 7.8.2. |
| Classifications[] | Optional | See sections 7.7 and 7.8.2. |

## 10.2 CIM_InstalledSoftwareIdentity

CIM_InstalledSoftwareIdentity is used to associate an instance of CIM_System and an instance of CIM_SoftwareIdentity. CIM_InstalledSoftwareIdentity is conditional and shall be implemented when Installed Software is modeled. Table 13 contains the requirements for elements of this class.

**Table 13 – Class: CIM_InstalledSoftwareIdentity**

| Elements | Requirement | Notes |
|---|---|---|
| System | Mandatory | **Key:** This property is a reference to the Scoping Instance.<br>Cardinality * |
| InstalledSoftware | Mandatory | **Key:** This property is a reference to the Software Identity that represents Installed Software.<br>Cardinality * |

## 10.3 CIM_ElementSoftwareIdentity

CIM_ElementSoftwareIdentity is used to associate an instance of CIM_ManagedElement and an instance of CIM_SoftwareIdentity when the instance of CIM_ManagedElement is instrumented. Table 14 contains the requirements for elements of this class.

**Table 14 – Class: CIM_ElementSoftwareIdentity**

| Elements | Requirement | Notes |
|---|---|---|
| Antecedent | Mandatory | **Key:** This property is a reference to the Software Identity.<br>Cardinality * |
| Dependent | Mandatory | **Key:** This property is a reference to the instance of CIM_ManagedElement.<br>Cardinality * |
| ElementSoftwareStatus | Mandatory | See section 7.4.1.1.1. |

## 10.4 CIM_SystemSpecificCollection

CIM_SystemSpecificCollection is used to represent a collection of Available Software. Table 15 contains the requirements for elements of this class.

**Table 15 – Class: CIM_SystemSpecificCollection**

| Elements | Requirement | Notes |
|---|---|---|
| InstanceID | Mandatory | **Key** |
| ElementName | Mandatory | See section 7.6.1. |

## 10.5 CIM_HostedCollection

CIM_HostedCollection is used to associate CIM_System and CIM_SystemSpecificCollection. CIM_HostedCollection is conditional and shall be implemented when an instance of CIM_SystemSpecificCollection is instrumented. Table 16 contains the requirements for elements of this class.

**Table 16 – Class: CIM_HostedCollection**

| Elements | Requirement | Notes |
|---|---|---|
| OwningElement | Mandatory | **Key:** This property is a reference to the Scoping Instance.<br>Cardinality 1 |
| OwnedElement | Mandatory | **Key:** This property is a reference to the collection of Available Software.<br>Cardinality 0..1 |

## 10.6 CIM_MemberOfCollection

CIM_MemberOfCollection is used to associate an instance of CIM_SystemSpecificCollection and an instance of CIM_SoftwareIdentity. CIM_MemberOfCollection is conditional and shall be implemented when an instance of CIM_SystemSpecificCollection is instrumented. Table 17 contains the requirements for elements of this class.

**Table 17 – Class: CIM_MemberOfCollection**

| Elements | Requirement | Notes |
|---|---|---|
| Collection | Mandatory | **Key:** This property is a reference to the collection of Available Software.<br>Cardinality * |
| Member | Mandatory | **Key:** This property is a reference to the instance of CIM_SoftwareIdentity that represents an Available Software.<br>Cardinality * |

**EXPERIMENTAL**

## 10.7 CIM_SoftwareIdentityResource

CIM_SoftwareIdentityResource is used to represent the location of a Software Identity, which could be used as input to the software installation service (see the *Software Update Profile*). Table 18 contains the requirements for elements of this class.

**Table 18 – Class: CIM_SoftwareIdentityResource**

| Elements | Requirement | Notes |
|---|---|---|
| SystemCreationClassName | Mandatory | **Key** |
| SystemName | Mandatory | **Key** |
| CreationClassName | Mandatory | **Key** |
| Name | Mandatory | **Key** |
| InfoFormat | Mandatory | |
| AccessInfo | Mandatory | |
| ResourceType | Optional | |

**EXPERIMENTAL**

## 10.8 CIM_SAPAvailableForElement

CIM_SAPAvailableForElement is used to associate CIM_SoftwareIdentityResource and CIM_SoftwareIdentity. CIM_SAPAvailableForElement is conditional and shall be implemented when the location information of CIM_SoftwareIdentity is represented. Table 19 contains the requirements for elements of this class.

**Table 19 – Class: CIM_SAPAvailableForElement**

| Elements | Requirement | Notes |
|---|---|---|
| AvailableSAP | Mandatory | **Key:** This property is a reference to the CIM_SoftwareIdentityResource instance.<br>Cardinality 1 |
| ManagedElement | Mandatory | **Key:** This property is a reference to the Software Identity.<br>Cardinality 0..1 |

## 10.9 CIM_HostedAccessPoint

CIM_HostedAccessPoint is used to associate CIM_System and CIM_SoftwareIdentityResource when an instance of CIM_SoftwareIdentityResource is instrumented. Table 20 contains the requirements for elements of this class.

**Table 20 – Class: CIM_HostedAccessPoint**

| Elements | Requirement | Notes |
|---|---|---|
| Antecedent | Mandatory | **Key:** This property is a reference to the scoping CIM_System instance.<br>Cardinality 1 |
| Dependent | Mandatory | **Key:** This property is a reference to instance of CIM_SoftwareIdentityResource.<br>Cardinality * |

**EXPERIMENTAL**

## 10.10 CIM_OrderedComponent

CIM_OrderedComponent is used to associate an instance of CIM_SoftwareIdentity that represents a Software Bundle and an instance of CIM_SoftwareIdentity that represents one of the discrete software images contained in the Software Bundle. Table 21 contains the requirements for elements of this class.

**Table 21 – Class: CIM_OrderedComponent**

| Elements | Requirement | Notes |
|---|---|---|
| GroupComponent | Mandatory | **Key:** See section 7.7.1.<br>Cardinality * |
| PartComponent | Mandatory | **Key:** See section 7.7.2.<br>Cardinality * |
| AssignedSequence | Mandatory | See section 7.7.3. |

**EXPERIMENTAL**

**EXPERIMENTAL**

## 10.11 CIM_OrderedDependency

CIM_OrderedDependency is used to associate an instance of CIM_SoftwareIdentity that represents an Installation Dependency and an instance of CIM_SoftwareIdentity for which the Installation Dependencies are represented. Table 22 contains the requirements for elements of this class.

**Table 22 – Class: CIM_OrderedDependency**

| Elements | Requirement | Notes |
|---|---|---|
| Antecedent | Mandatory | **Key:** See section 7.9.1.1.<br>Cardinality * |
| Dependent | Mandatory | **Key:** See section 7.9.1.2.<br>Cardinality * |
| AssignedSequence | Mandatory | See section 7.9.1.3. |

**EXPERIMENTAL**

## 10.12 CIM_RegisteredProfile

The CIM_RegisteredProfile class is defined by the *Profile Registration Profile*. The requirements denoted in Table 23 are in addition to those mandated by the *Profile Registration Profile*.

**Table 23 – Class: CIM_RegisteredProfile**

| Elements | Requirement | Notes |
|---|---|---|
| RegisteredName | Mandatory | This property shall have a value of "Software Inventory". |
| RegisteredVersion | Mandatory | This property shall have a value of "1.0.0". |
| RegisteredOrganization | Mandatory | This property shall have a value of 2 (DMTF). |

NOTE: Previous versions of this document included the suffix "Profile" for the RegisteredName value. If implementations querying for the RegisteredName value find the suffix "Profile", they should ignore the suffix, with any surrounding white spaces, before any comparison is done with the value as specified in this document.

# ANNEX A
# (informative)

# Change Log

| Version | Date | Description |
|---|---|---|
| 1.0.0a | 2006/10/19 | Preliminary Standard version. |
| 1.0.0b | 2007/03/12 | Preliminary Standard refresh. Minor errata incorporation. |
| 1.0.0 | 2007/11/21 | Final Standard |
| 1.0.0 | 2008/07/21 | Section 7.7, 7.9, 9.1.11, 9.1.12, 9.1.13, 10.7. 10.10, 10.11 marked EXPERIMENTAL in order to go FINAL |
| 1.0.0 | 2008/07/28 | Address David Hines ballot comments<br>http://www.dmtf.org/apps/org/workgroup/platformprofiles/download.php/39476/DSP1023_comments_Hines_2008-07-28.xls |

# ANNEX B
# (informative)

## Acknowledgments

The authors wish to acknowledge the following people.

**Editor:**

- RadhaKrishna R. Dasari – Dell, Inc.**Contributors:**
- RadhaKrishna R. Dasari – Dell, Inc.Jon Hass – Dell, Inc.
- Khachatur Papanyan – Dell Inc.
- Aaron Merkin – IBM
- Jeff Hilland – Hewlett-Packard Corporation
- Christina Shaw – Hewlett-Packard Corporation
- Michael Tehranian – Sun Microsystems
- Perry G. Vincent – Intel Corporation
- John Leung – Intel Corporation