
# CIM Device Model White Paper

## CIM Version 2.7

## Version 0.9 June 19, 2003

## Abstract

The DMTF Common Information Model (CIM) is a conceptual information model for describing computing and business entities in Internet, enterprise and service provider environments. It provides a consistent definition and structure of data, using object-oriented techniques. The CIM Schema establishes a common conceptual framework that describes the managed environment.

This document reviews the concepts that are currently modeled in the CIM 2.7 Device Model. This paper mirrors the organization of the classes as they are presented in the MOF and UML/Visio diagrams.

# Notice

***DSP0144*** ***Status: Work-In-Progress***



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

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patent owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified after publication, and shall be indemnified and held harmless by any party implementing the standard from any and all claims of infringement by a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

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

# Table of Contents

# 1 Introduction

## 1.1 Overview

The CIM Device Common Model describes the functionality provided by hardware, as well as providing configuration and state data. The model covers a wide breadth of hardware. It addresses low-level concepts such as sensors, batteries and fans, and high-level abstractions such as Storage Volumes.

There are several important concepts related to a CIM_LogicalDevice:

- Devices represent the abstract concepts of the functionality, configuration and state of hardware. They have a "Realized" relationship referencing the hardware that they describe.
- Typically, a single hardware component provides multiple functionalities that are realized as multiple different LogicalDevices.
- The configuration of the underlying hardware and software is critical to managing the device.
- The interaction between the various devices (i.e., their interconnections) can play a crucial role in managing the device itself.
- Devices are described as components of a CIM_System that contain them. This relationship is described by the mandatory SystemDevice relationship. It has been identified that this single level of containment makes it difficult to describe a device that is contained by another element, other than a system.

The goal of this paper is to review the concepts that are currently modeled in the CIM 2.7 Device Model. This paper mirrors the organization of the classes as they are presented in the MOF and UML/Visio diagrams.

# 2 The Device Model

## 2.1 Background and Assumptions

It is assumed that the reader is familiar with the concepts and terminology presented in the CIM Specification, CIM Concepts Paper, and CIM Core Paper.

## 2.2 Conceptual Areas Addressed by the Model

The Device Model establishes a basic classification of the functionality of networking and compute hardware entities and their associations. The Device Model is broken down into the following conceptual areas, which align with the Device MOF:

- Device Elements (Overview)
- Cooling and Power
- Processor
- Controller
- Ports
- Network Adapter
- Fibre Channel
- InfiniBand
- Storage Devices
- Storage Extents
- SCC Extents
- Storage Services
- Storage Library
- User Devices
- Memory
- Modems
- Printing
- Sensors
- USB

In addition, each conceptual area is broken down into individual classes and associations.

Note: All classes defined by the DMTF in the Device Models are named using the following syntax: <schema name>_<class name>. An underscore is used as a delimiter

between the <schema name> and the <class name>. In this case the schema name is 'CIM'. For reading convenience, the CIM_ prefix is omitted on class names throughout this paper. For clarity, class names are usually identified by ***bold italics***.

## 2.3 Device Elements

The Device Elements MOF defines the top level associations of the Device Model, which are used or extended by the subsequent sections of the model. These associations include **Device Connection**, **Device Identity**, **Controlled By**, and **Device Software**.

The following sub-sections describe these associations. (Note that some of the descriptions are excepted from the book, *Common Information Model, Implementing the Object Model for Enterprise Management*[1].)

### *Logical Device*

LogicalDevice is defined in the Core Model. Since LogicalDevice is the root for the Device Model it is included here. It is important to remember that subclasses of logical device do not represent the hardware itself; but the functionality that the hardware provides.

### *Device Connection*

Connections between devices abound in the real world. A printer may be connected to a parallel port. A USB device may be connected to a port on a USB hub. These are examples of the CIM_DeviceConnection association. Typically, when devices communicate, they first negotiate their transmit speed and data width. This information is part of the class definition of CIM_DeviceConnection - the NegotiatedSpeed and NegotiatedDataWidth properties.

### *Controlled By*

A subclass of CIM_DeviceConnection is CIM_ControlledBy. The latter describes that a particular connection is actually between a CIM_Controller and its "downstream" device. Controllers are devices with a single protocol stack whose primary purpose is to communicate with, control and reset connected devices. There are many subclasses of CIM_Controller, addressing SCSI, PCI, USB, serial, parallel and video controllers.

### *Device Identity*

Relating different logical aspects of a single device is accomplished using CIM_DeviceIdentity. DeviceIdentity is a subclass of the CIM_LogicalIdentity relationship, defined in the Core Model. Each aspect is represented by a LogicalDevice

[1] Common Information Model, Implementing the Object Model for Enterprise Management by Winston Bumpus, John Sweitzer, Patrick Thompson, Andrea Westerinen, Ray Williams. Wiley Computer Publishing.

instance. The aspects are then tied together using the DeviceIdentity relationship. The association's semantics are similar to those conveyed by multiple inheritance (which is not supported by the CIM meta-model.)

## 2.4 Cooling and Power

The cooling and power MOF addresses the possible sources of power for devices (such as power supplies and/or batteries) and the possible sources for cooling (such as fans and pipes). There are specialized associations to tie a device to its appropriate power and/or cooling hardware. These associations are **Supplies Power**, **Associated Cooling**, and **Associated Battery**. Remember that the power and cooling objects also have a **System Device** relationship to their containing system.

## 2.5 Processor

The processor MOF addresses the devices that represent processors and watchdogs.

## 2.6 Controller

A controller manages communications (data, control, and reset) to 'downstream' devices, related as master/slave. It is dedicated to running an explicit protocol, such as PCI, USB, Video, Serial, Parallel, etc. Examples of controllers are **USB Controllers**, and **Serial Controllers**. The **Controlled By** relationship is a specialized version of **Device Connection**. It defines the relationship between the controller and the devices that are downstream.

## 2.7 Logical Port and Logical Port Group

**Logical Port** defines the classes describing communications ports, including Ethernet, Token Ring, and InfiniBand. This segment also includes the definitions of the associated statistics classes for each type of Logical Port.

**Logical Port Group** is defined to group ports for administrative purpose (e.g., groupings of ports indicating a Fibre Channel node, InfiniBand node, etc.). Logical Port Group is not intended to show the functional aspects of a physical card, but an administrative grouping.

## 2.8 Network Adapter

NetworkAdapter and its associated classes have been deprecated and replaced with by Network Port and its associated objects.

## 2.9 Fibre Channel

**FC Port** and **FC Port Statistics** extend from Network Port and Network Port Statistics respectively. Although not explicitly defined, the Fibre Channel Sub-Model uses Logical Port Group to define the Fibre Channel node.

FC zoning information is defined in subclasses of **Connectivity Collection** (from the Core Model). **ZoneSets** and **Zones** are defined. Membership in the zone is indicated by the class, **Zone Memebership Setting Data,** an extension of **Membership Setting Data** (from the Network Model). Lastly, **Named Address Collection** is used to represent a Zone Alias. Zoning control is managed through the **Zone Service**.

For further information regarding the Fibre Channel Sub-Model, see the Storage Management Initiative Specification which details the steps necessary to model most aspects of Fibre Channel SANs/Fabrics. [1]

## 2.10 InfiniBand

In CIM V2.8, the InifinBand (IB) Sub-Model parallels the Fibre Channel Sub-Model. So, **IB Ports** are subclassed from **NetworkPort** and the **IBPortStatistics** are also defined. Currently, only the **IB Subnet Manager** service exists in the CIM Schema. This service is hosted by an **Admin Domain** representing the IB fabric.

## 2.11 Storage Devices

Storage Devices represent the realization of physical media players (for example, hard drives, CD-ROM players, etc/). They are represented in the model as **Media Access Devices**, and have three distinct types of relationships:

- A relationship to the media relied on for data storage (**Media Present**)
- A relationship to the data transport mechanism utilized (**Controlled By**)
- A relationship to the physical objects which realize the logical devices (**Realizes**)

At a minimum, a storage device "player" should be instantiated as a PhysicalPackage, and the media should be instantiated as **Storage Extents**. A Realizes association would exist between the Physical Package and the device, to show the hardware behind the player. A Realizes association would exist between the **Physical Media** and the **Storage Extent** to show the relationship between the logical and physical "media".

For non-removable devices, an optimization can be made since the same Physical Package realizes both the "player" and "media" functionality.

Lastly, the Physical Package could model its connectors as **Physical Connectors**. The aggregate association **Connector On Package** is used to show the relationship between the connector and the package.

## 2.12 Storage Extents

Storage Extents represent the realization of the physical medium (e.g., the. disk, CD-ROM, tape, floppy, memory, etc.) and describe the characteristics of a storage space. They also describe the composition of several storage spaces into a new data storage space, or the partitioning of a large medium into multiple data storage spaces.

A Storage Extent can have several distinct relationships. It can have relationships to:

- The Media Access Device (via **Media Present**, as discussed above)

- Physical objects (via **Realizes,** also discussed above)
- Another Storage Extent (via **Based On**)
- The file system (**Resides On Extent**)
- A defect (**Storage Defect**)
- A redundancy group (**Extent Redundancy Component** or **Acts As Spare**).

For more information regarding Storage Extents, see the Device Storage Sub-Model White Paper (DSP0137).

## 2.13 SCC Extents

The classes based on the SCC Model have been deprecated and replaced with using **Storage Extent** and its subclass **Composite Extent.** The associations referencing the SCC- based classes have also been deprecated and replaced with **Based On**, **Composite Extent Based On**, and **Protected Extent Based On**.

## 2.14 Storage Services

The Storage Services MOF defines two distinct services:

- **Storage Configuration Service** is used to create and manipulate **Storage Volumes** and **Storage Pools**. The service is tied to its capabilities (**Storage Capabilities**) using the **Element Capabilities** association. The methods within this service take instances of **Storage Settings** as input, to determine how the storage should be created or manipulated.
- The **Configuration Reporting Service** outputs the current storage configuration and potential for growth. The service may be used in several circumstances:
  - To report growth potential ("how many can I have?")
  - To report information on objects not directly modeled for performance or other reasons. (i.e., the capacity of thousands of drives in a storage subsystem)
  - To report counts of 'things' or counts of 'units' (i.e., the number of disk drives could be reported or the capacity that they provide)

## 2.15 Storage Library

Storage Library devices consist of **Label Readers** and **Media Transfer Devices** (**Pickers**, **Changers**, and **Access Ports**). The specialized relationships used within the Storage Library are **Associated Label Reader**, **Access Label Reader**, **Picker Label Reader**, **Picker For Changer**, and **Library Exchange**.

## 2.16 User Devices

The user device MOF addresses hardware related to user interaction. This hardware includes keyboards, displays, point devices, scanners, and doors. The **Device Connection** or **Controlled By** associations can be used between a user device and its underlying controlling interface (USB, VESA, PCI, etc.).

## 2.17 Memory

Memory is a specialization of **Storage Extent.**. Memory is anything from cache to a RAM Disk. The further specialization of memory as **Volatile Storage**, ***NonVolatile Storage***, and **Cache Memory** (processor cache) are being deprecated and simplified in CIM V2.8.

## 2.18 Modems

The modems MOF defines **Call Based Modem** and **Connection Based Modem**. It also defines out of band alerting for modems via the association, **OOB Service On Modem** to the **OOB Service,** and wake up services via the **Wake Up Service On Modem** association to the **Wake Up Service**.

## 2.19 Printing

The printer segment addresses the area of printers, print services, print jobs and queues, and print access points. Several MIBs have been defined to provide a standard means of instrumenting printers, their jobs and the various printing options. These include:

- Printmib-mib-info (draft-ietf-printmib-finishing-16.txt)
- Printmib-finishing (draft-ietf-print-mib-info-15.txt)

Data from these MIBs can be mapped into the CIM hierarchy. Their mapping is indicated using the MappingStrings qualifier on a property.

Any model for the management of a printer and its related infrastructure must take into account the different types of printing topologies that might be deployed within an enterprise. The simplest scenario consists of a printer that is directly attached to a computer system. It is not networked, only supports a single user at a time and is not running any spooling software. Adding a print spooler may provide additional management capabilities for a print job such as job prioritization. More complex scenarios arise when systems are networked together. In such cases each client may be able to manage print jobs as well as the networked entity that is controlling the print mechanism. In other deployments, the printer may be a “black box” where it simply provides an IP address to which clients can send jobs that are to be output, or the printer may be another instance of a generic computer system with a directly attached printer.

The model is a logical representation for printing. It is not intended to cover the physical aspects of a printer such as the form factor for any of the printer components, asset information and power information. The Physical Model within CIM already provides the necessary objects and associations that allow this type of information to be specified.

## 2.20 Sensors

- Sensor instrumentation providers should directly use the **Sensor** and **Numeric Sensor** classes, and should not define subclasses to create new Sensor types. This should keep the model flexible for the sensors with varying capabilities. The **Associated Sensor** relationship is used to tie a device to its sensor.
- The **Alarm Device** covers audible, visible, and motion alarms. A visible alarm covers something as simple as an LED indicator. The **Associated Alarm** relationship is used to tie a device to its alarm.

## 2.21 USB

The USB classes include:

- **USB Port,** subclassed from **Logical Port**
- **USB Device** to represent a generic USB device
- **USB Hub,** subclassed from USB Device
- **USB Controller,** which subclasses from **Controller**

Also, threee associations exist specifically for USB:

- **USB Port On Hub** which subclasses from **Port On Device** and has a cardinality requiring at least one **USB Port**
- **USB Controller Has Hub** which has a cardinality requiring one **USB Controller** and at least one **USB Hub**
- **USB Connection** which subclasses from **Device Connection** and requires a 1:1 relationship between a **USB Port** and **USB Device**

# 3 Relationships to Other Standards and Specifications

## 3.1 Overlapping Standards and Specifications

DMTF DMI MIFs (Management Information Format files, available at http://www.dmtf.org/standards/standard_dmi.php)

IETF SNMP MIBs (Management Information Bases, available at http://www.ietf.org/)

ANSI T10 Standards (available at http://www.t10.org/)

ANSI T11 Standards (available at http://www.t11.org/)

InfiniBand Architecture Specifications (available at http://www.infinibandta.org/specs)

SNIA Storage Management Initiative Specification (available at http://www.snia.org/smi/tech_activities/smi_spec_pr/spec)

## 3.2 A Mapping of DMI MIFs into CIM

The MappingStrings qualifiers are used to indicate how the CIM class property's value correlates to the DMI attribute. For example,

```
MappingStrings {"MIF.DMTF|System Hardware Security|001.4"}
```

shows that the property correlates to version 4 of the DMI "System Hardware Security" group's attribute ID # 1.

Therefore, the MIF mapping string takes on the form:

```
MappingStrings {"MIF.DMTF|<DMI group name>|<attribute
ID>.<group version #>"}
```

## 3.3 A Mapping of SNMP MIBs into the Model

The MappingStrings qualifiers are used to indicate how the CIM class property's value correlates to the SNMP attribute. For example,

```
MappingStrings {"MIB.IETF|MIB-II.sysServices"}
```

shows that the property correlates to the sysServices object in the IETF's MIB-II mib definition.

Therefore, the MIB mapping string takes on the form:

```
MappingStrings {"MIB.IETF|<IETF MIB Name>.<object name>"}
```

## 3.4 A Mapping of IBTA MADs into the Model

The MappingStrings qualifiers are used to indicate how the CIM class property's value correlates to the InfiniBand Trade Association's MAD attribute. For example,

```
MappingStrings {"MAD.IBTA | SMInfo | Priority"}
```

shows that the property correlates to the SMInfo MAD in the InfiniBand Architecture Specification.

Therefore, the MIB mapping string takes on the form:

```
MappingStrings {"MAD.IBTA|<IBTA MAD Name>.<attribute name>"}
```

## 3.5 A Mapping of ANSI T11 Data Structures into the Model

The MappingStrings qualifiers are used to indicate how the CIM class property's value correlates to the ANSI data structure. For example,

```
MappingStrings{"FC-
SWAPI.T11|ErrorCounters|1.SWAPI_STATS_LBL_FBSY"}]
```

shows that the property correlates to the Error Counters data structure in the ANSI Standard.

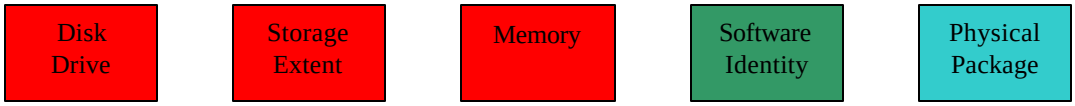
Therefore, the data structure mapping string takes on the form:

```
MappingStrings {"<Specification>.T11|<Data
Structure>|<Version>.<Name>"}
```

# 4 Device Model Use Case

## 4.1 Disk Drive Example:

A single example can not cover the Device Model in its entirety – because its scope is too large, addressing all the various aspects of hardware functionality, configuration and state. In fact, the Device Model can be broken down to individual components (cooling and power, processors, storage, etc.) that are managed individually. So, to understand the model, a specific, rather common example is chosen – that of a disk drive.



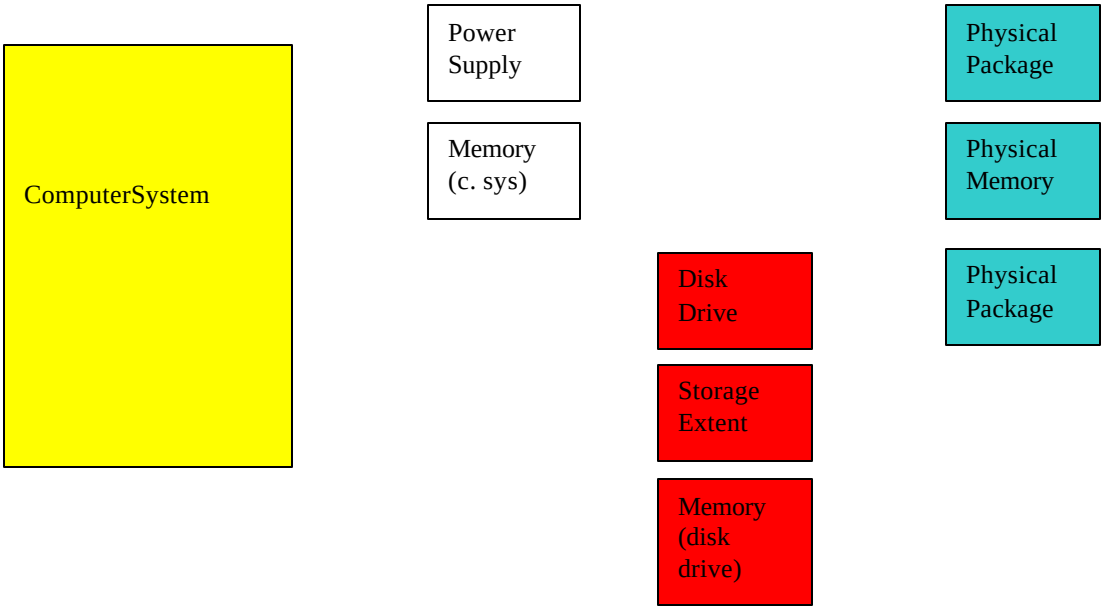

The functionality that we typically associate with a disk drive includes the:

- Physical Package, which represents the drive mechanism that you can see and touch – containing storage, the read/write hardware, on-board flash or EPROMs, etc.
- Disk Drive, which represents the functionality to read/write data from the medium – realized as a type of Media Access Device.
- Storage Extent, which represents the functionality of the medium used to storage the data. It may or may not be removable.
- Memory, which represents the internal cache buffers.
- Software Identity, which represents the firmware and device driver code that is available for the drive.

Then, there are various associations that tie these classes and concepts together:

- The Media Present association is used to tie the Storage Extent to its Disk Drive.
- The Associated Memory association is used to tie the Memory to its Disk Drive.
- The Device Software Identity association (defined in CIM V2.8) is used to tie the Software Identity to its Disk Drive
- The Realizes association is used to tie the Disk Drive, Storage Extent, and Memory to the Physical Package.

It is practical to manage a LogicalDevice in the context of the System in which it is functioning. Therefore, the next step in the example is to place the disk drive in the context of a Computer System.




In this example:

- The ComputerSystem has a SystemDevice relationship to:
  - Power Supply
  - Memory (for the computer system)
  - Disk Drive
  - Storage Extent
  - Memory (for the disk drive)

    It is cumbersome that the Memory for the disk drive is a component of the Computer System versus a component of the disk drive. However, the Memory is associated to the Disk Drive using the Associated Memory relationship. This indicates that the Memory is indeed "dedicated" to the drive.

- The Disk Drive has the following associations:
  - System Device to describe its component relationship to the Computer System
  - Media Present to describe the dependent relationship with Storage Extent (its medium).
  - Associated Memory to describe its usage of Memory.
  - Realizes to tie to the Physical Package (hardware).

# 5 Future Work

There are several areas of work planned for CIM V2.8 and following releases:

- Clean up work on Memory (introducing a generic cache association versus a specific processor cache subclass)
- Clean up and simplification of the Sensor Sub-Model
- Addition of methods for zone control to Zone Service
- Introduction of Protocol Controller and the deprecation of SCSI Controller
- Support for emerging technologies such as iSCSI.
- Support for copy services for point-in-time copies

# Appendix A – Change History

| | | |
|---|---|---|
| Version 0.9 | June 19, 2003 | Initial Draft |

# Appendix B – References

CIM Core and Common Models - Versions 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, and 2.7 - Downloadable from http://www.dmtf.org/standards/standard_cim.php

Common Information Model (CIM) Specification, V2.2, June 14, 1999 - Downloadable from http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf

DMTF Specifications - Approved Errata - Downloadable from http://www.dmtf.org/standards/standard_cim.php

Unified Modeling Language (UML) from the Open Management Group (OMG) - Downloadable from http://www.omg.org/uml/

Internet Engineering Task Force (IETF) - MIBs and Work Group information at http://www.ietf.org

Core White Paper, DSP111, June 2003 – Downloadable from http://www.dmtf.org/standards/published_documents.php

System White Paper, DSP151, June 2003 – Downloadable from http://www.dmtf.org/standards/published_documents.php