Specification

DSP0102

LDAP CIM Physical Schema v2.3 Mapping

May 8th, 2000

Abstract

This draft presents a LDAP schema for the DMTF CIM Physical model version 2.3 [3].

Table of Contents

0. Draft Changes

0.1 Initial version

Changed numbering and updated to physical 2.3 model. To do issues include adding DIT consideration section, OIDs, and OID assignment appendix. Class mapping issues include how to handle AdapterActiveConnection, ComputerSystemPackage, LibraryPackage, PackageCooling, and PackageAlarm.

0.2 Changes to 3/28 version

The 3/28 version has additional tables and internal links added for navigation. In addition, sections 2.4 and 2.5 added to list attribute/OIDs and object classes/OIDs from the core mapping used in this document. Draft text has been added to Section 2.1.

1. Introduction

This draft presents a LDAPv3 [1,2] schema for the DMTF CIM Physical model. Associations are mapped using a combination of auxiliary classes and DIT structure rules.

All attribute, object class, and name form OIDs are placeholders. Further, structure rule identifiers are placeholders and should be replaced as dictated by local implementations.

In the mapping of properties to attributes, syntax object identifiers have been replaced by textual names. The correspondence between names and OIDs is shown in the following table:

Textual Name	OID
Boolean	1.3.6.1.4.1.1466.115.121.1.7
DN	1.3.6.1.4.1.1466.115.121.1.12
DirectoryString	1.3.6.1.4.1.1466.115.121.1.15
Integer	1.3.6.1.4.1.1466.115.121.1.27

2. LDAP Mapping Considerations

There are several special considerations in mapping the physical model from CIM to LDAP. They are discussed in this section.

There are some classes that aren't included in this mapping: Since MediaTransferDevice isn't included in the device model, DeviceServiceLocation isn't included here. MediaPhysicalStatInfo isn't included here because it is filled with nothing but counters. Without StorageExtents, the associations RealizesExtent, RealizesPExtent, RealizesDiskPartition, RealizesAggregatePExtent, and RealizesTapePartition do not make sense to include here. Finally, the PackageTempSensor association is not included because there are no TemperatureSensors in the schema.

2.1 DIT Considerations

This mapping is concerned with CIM classes derived from CIM_PhysicalElement and associations relating instances of these classes with other classes. Instances of subclasses of CIM_PhysicalElement are identified by values of the two keys CreationClassName and Tag. This approach produces a flat, non hierarchical, namespace of physical

elements. CIM does not use weak associations to structure the namespace of physical elements because the relationship between physical elements may change, e.g., a card may be removed from one slot and placed in another. Were the card to be placed in a weak association to the slot, its keys would not be well-defined during the transition and would have to be changed after the move.

It is suggested that the RDN for instances of non abstract subclasses cimPhysicalElement be a value of orderedCimKeys. The value should be taken from the CIM key properties CreationClassName and Tag. This RDN value will provide for unambiguous, unchanging identification of the physical element, at least for elements from the same CIM namespace.

Other than RDN values, this mapping does not specify any DIT structure for the mapping of CIM physical element instances. As flat namespaces are normally not desirable in directories, some opportunities for site-specific DIT structures are suggested below.

Physical elements are typically constituents of a system of some kind. In CIM, the system composition is expressed by the CIM_SystemComponent aggregation. Although it is conceivable that a physical element is a component of more than one system, e.g., an admin domain and a unitary computer system, it is likely that one system will be most appropriate as a root for DIT containment. In this case the SystemComponent aggregation can be mapped, not using the auxiliary class approach of the CIM core model mapping, but as DIT containment, the physical element instances being immediately subordinate to the system instance. It is also possible to use both approaches together, that is, subordinating physical elements to a system instance, and to represent the component relationship using cim22SystemComponentAuxClass.

Another opportunity for deriving DIT structure from CIM relationships stems from the association CIM_Container and its subclasses: CIM_ChassisInRack, CIM_PackageInChassis, CIM_CardOnCard, CIM_PackagedComponent and CIM_ConnectorOnPackage.

2.2 Helper Classes for Indexed Arrays

This section presents all of the helper classes that are defined to support mapping of indexed arrays in CIM classes in the physical model.

2.2.1 cimServicePhilosophyInstance

The class cim23PhysicalFrame defines two linked indexed arrays: ServicePhilosophy and ServiceDescription. These are replaced with separate instances of cimServicePhilosophyInstance, which are DIT contained by cim23PhysicalFrame.

```
( <oid-at74> NAME 'cimServicePhilosophy'
  DESC 'ServicePhilosophy is an enumerated, integer value that
      indicates whether the Frame is serviced from the top
      (value=2), front (3), back (4) or side (5), whether it has
      sliding trays (6) or removable sides (7), and/or whether
      the Frame is moveable (8), for example, having rollers.
      Values are 0="Unknown", 1="Other", 2="Service From Top",
```

```
      3="Service From Front", 4="Service From Back", 5="Service From
      Side", 6="Sliding Trays", 7="Removable Sides", 8="Moveable"'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at75> NAME 'cimServiceDescriptions'
  DESC 'A free-form strings providing more detailed explanations
      for this entries.
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-oc45> NAME 'cimServicePhilosophyInstance'
  DESC 'helper class to tie ServicePhilosophy and
      ServiceDescriptions in PhysicalFrame together'
  SUP top
  MUST ( arrayIndex )
  MAY ( cimServicePhilosophy $ cimServiceDescription )
)

( <oid-nf10> NAME 'cimServicePhilosophyInstanceNameForm'
  OC cimServicePhilosophyInstance
  MUST ( arrayIndex )
)

( <sr10> NAME 'cimServicePhilosophyInstanceStructureRule'
  FORM cimServicePhilosophyInstanceNameForm
  SUP ( <sr24> )
)
```

2.2.2 cimChassisTypeInstance

The class cim23Chassis defines two linked indexed arrays: ChassisTypes and TypeDescriptions. In the LDAP mapping, these are replaced with separate instances of cimChassisTypeInstance, which are DIT contained by cim23Chassis.

```
( <oid-at76> NAME 'cimChassisTypes'
  DESC 'An enumerated, integer value indicating the type of
      Chassis. Values: 1="Other", 2="Unknown", 3="Desktop",
      4="Low Profile Desktop", 5="Pizza Box", 6="Mini Tower",
      7="Tower", 8="Portable", 9="LapTop", 10="Notebook",
      11="Hand Held", 12="Docking Station", 13="All in One",
      14="Sub Notebook", 15="Space-Saving", 16="Lunch Box",
      17="Main System Chassis", 18="Expansion Chassis",
      19="SubChassis", 20="Bus Expansion Chassis",
      21="Peripheral Chassis", 22="Storage Chassis",
      23="Rack Mount Chassis", 24="Sealed-Case PC"'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at77> NAME 'cimTypeDescriptions'
  DESC 'A free-form strings providing more information on the
      ChassisTypes array entries.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-oc46> NAME 'cimChassisTypeInstance'
  DESC 'helper class to tie ChassisType and
      TypeDescriptions in Chassis together'
  SUP top
  MUST ( arrayIndex )
  MAY ( cimChassisType $ cimTypeDescription )
)
```

```
( <oid-nf11> NAME 'cimChassisTypeInstanceNameForm'
  OC cimChassisTypeInstance
  MUST ( arrayIndex )
)

( <sr11> NAME 'cimChassisTypeInstanceStructureRule'
  FORM cimChassisTypeInstanceNameForm
  SUP ( <sr26> )
)
```

2.2.3 cimMediaTypesSupportedInstance

The class cim23StorageMediaLocation defines two linked indexed arrays: MediaTypesSupported and MediaSizesSupported. In the LDAP mapping, these are replaced with separate instances of cimMediaTypeSupportedInstance, which are DIT contained by cim23StorageMediaLocation.

```
( <oid-at78> NAME 'cimMediaTypesSupported'
  DESC 'Certain StorageMediaLocations may only be able to accept a
      limited set of PhysicalMedia MediaTypes. This property defines
      the types of Media that are acceptable for placement in the
      Location. Values are 0="Unknown", 1="Other", 2="Tape
      Cartridge", 3="QIC Cartridge", 4="AIT Cartridge", 5="DTF
      Cartridge", 6="DAT Cartridge", 7="8mm Tape Cartridge",
      8="19mm Tape Cartridge", 9="DLT Cartridge", 10="Half-Inch
      Magnetic Tape Cartridge", 11="Cartridge Disk", 12="JAZ
      Disk", 13="ZIP Disk", 14="SyQuest Disk", 15="Winchester
      Removable Disk", 16="CD-ROM", 17="CD-ROM/XA", 18="CD-I",
      19="CD Recordable", 20="WORM", 21="Magneto-Optical",
      22="DVD", 23="DVD-RW+", 24="DVD-RAM", 25="DVD-ROM",
      26="DVD-Video", 27="Divx", 28="Floppy/Diskette", 29="Hard
      Disk", 30="Memory Card", 31="Hard Copy", 32="Clik Disk",
      33="CD-RW", 34="CD-DA", 35="CD+", 36="DVD Recordable",
      36="DVD-RW", 37="DVD-Audio", 38="DVD-5", 39="DVD-9",
      40="DVD-10", 41="DVD-18", 42="Magneto-Optical Rewriteable",
      43="Magneto-Optical Write Once", 44="Magneto-Optical
      Rewriteable (LIMDOW)", 45="Phase Change Write Once",
      46="Phase Change Rewriteable", 47="Phase Change Dual
      Rewriteable", 48="Ablative Write Once", 49="Near Field
      Recording", 50="MiniQic", 51="Travan", 52="8mm Metal
      Particle", 53="8mm Advanced Metal Evaporate", 54="NCTP",
      55="LTO Ultrium", 56="LTO Accelis", 57="9 Track Tape",
      58="18 Track Tape", 59="36 Track Tape", 60="Magstar 3590",
      61="Magstar MP", 62="D2 Tape", 63="Tape - DST Small",
      64="Tape - DST Medium", 65="Tape - DST Large".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at79> NAME 'cimMediaSizesSupported'
  DESC 'The size (in inches) of the particular MediaTypes that may
      be placed in the Location.'
  SUP cim23Float32 SINGLE-VALUE
)

( <oid-oc47> NAME 'cimMediaTypesSupportedInstance'
  DESC 'helper class to tie MediaTypesSupported and
      MediaSizesSupported in StorageMediaLocation together'
  SUP top
  MUST ( arrayIndex )
  MAY ( cimMediaTypesSupported $ cimMediaSizesSupported )
)

( <oid-nf12> NAME 'cimMediaTypesSupportedInstanceNameForm'
```

```
  OC cimMediaTypesSupportedInstance
  MUST ( arrayIndex )
)

( <sr12> NAME 'cimMediaTypesSupportedInstanceStructureRule'
  FORM cimMediaTypesSupportedInstanceNameForm
  SUP ( <sr28> )
)
```

2.2.4 cimPhysicalLabelsInstance

The class cim23PhysicalMedia defines three linked indexed arrays: PhysicalLabels, LabelStates, and LabelFormats. In the LDAP mapping, these are replaced with separate instances of cimPhysicalLabelsInstance, which are DIT contained by cim23PhysicalMedia.

```
( <oid-at80> NAME 'cimPhysicalLabels'
  DESC '"labels" on the PhysicalMedia. The format of the labels and
      their state (readable, unreadable, or upside-down) is
      indicated in LabelFormats and LabelStates properties.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at81> NAME 'cimLabelStates'
  DESC 'An enumerated integer describing the state of a label on a
      PhysicalMedia. The Label is listed in the PhysicalLabels
      property.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at82> NAME 'cimLabelFormats'
  DESC 'An  enumerated integer describing the format of a label on
      a PhysicalMedia. The Labels is listed in the PhysicalLabels
      property.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc48> NAME 'cimPhysicalLabelsInstance'
  DESC 'helper class to tie PhysicalLabels, LabelStates, and
      LabelFormats in PhsyicalMedia together'
  SUP top
  MUST ( arrayIndex )
  MAY ( cimPhysicalLabels $ cimLabelStates $ cimLabelFormats )
)

( <oid-nf13> NAME 'cimPhysicalLabelsInstanceNameForm'
  OC cimPhysicalLabelsInstance
  MUST ( arrayIndex )
)

( <sr13> NAME 'cimPhysicalLabelsInstanceStructureRule'
  FORM cimPhysicalLabelsInstanceNameForm
  SUP ( <sr32> )
)
```

2.3 Syntax Conversion

In addition to the syntax conversion discussed in [4], the physical model has some attributes that require mapping floating point attributes. Mapping of these attributes is

accomplished by inheriting from the attributes cimFloat32 and cimFloat64, defined in [5]. Interested readers are directed there for information about these attribute definitions.

2.4 Attributes Defined in Core Mapping

The following table lists the attributes/OIDs used in this mapping that are defined in the core mapping document [4], the mapping guidelines document [5] or elsewhere.

OID	Attribute
1.3.6.1.4.1.412.100.1.2.5	arrayIndex
1.3.6.1.4.1.412.100.2.2.4	cimName
1.3.6.1.4.1.412.100.1.2.1	orderedCimKeys
1.3.6.1.4.1.412.100.1.2.2	orderedCimModelPath
1.3.6.1.4.1.412.100.2.2.26	cimElementRef
2.5.4.6	c
1.3.6.1.4.1.412.100.1.2.6	cimFloat32
1.3.6.1.4.1.412.100.1.2.7	cimFloat64

2.5 Object Classes defined in core mapping

The following table lists the object classes/OIDs used in this mapping that are defined in the core mapping document [4].

OID	Object Class
1.3.6.1.4.1.412.100.2.1.2.44	cim23ManagedElement
1.3.6.1.4.1.412.100.2.1.1.29	cim22ComponentAuxClass
1.3.6.1.4.1.412.100.2.1.1.22	cim22DependencyAuxClass
1.3.6.1.4.1.412.100.2.1.2.5	cim23PhysicalElement
1.3.6.1.4.1.412.100.2.1.2.46	cim23MemberOfCollectionAuxClass
1.3.6.1.4.1.412.100.2.1.1.28	cim22RealizesAuxClass
1.3.6.1.4.1.412.100.2.1.1.40	cim22ProductPhysicalElementsAuxClass
1.3.6.1.4.1.412.100.2.1.1.41	cim22FRUPhysicalElementsAuxClass
1.3.6.1.4.1.412.100.2.1.2.3	cim23CollectedMSEsAuxClass
1.3.6.1.4.1.412.100.2.1.1.27	cim22ProvidesServiceToElementAuxClass
1.3.6.1.4.1.412.100.2.1.1.30	cim22SystemComponentAuxClass
1.3.6.1.4.1.412.100.2.1.1.10	cim22ElementConfigurationAuxClass
1.3.6.1.4.1.412.100.2.1.1.13	cim22ElementSettingAuxClass

3. Class Definitions

For efficiency in the LDAP representation, associations are specified as a combination of auxiliary classes and DIT structure rules. Attribute definitions for each class are presented with the object class. Other definitions are also provided when necessary.

3.1 cim23Location

Locations are the position and address of a PhysicalElement.

```
( <oid-at83> NAME 'cimPhysicalPosition'
     DESC 'Position is a free-form string indicating the placement of
         PhysicalElement. It can specify slot information on a
         HostingBoard, mounting site in a Cabinet, or latitude and
         longitude information, for example, from a GPS. It is part
         of the key of the Location object.'
     SYNTAX DirectoryString{256} SINGLE-VALUE
   )

   ( <oid-at84> NAME 'cimAddress'
     DESC 'Address is a free-form string indicating a street, building
         or other type of address for the PhysicalElement"s
         Location.'
     SYNTAX DirectoryString{1024} SINGLE-VALUE
   )

   ( <oid-oc49> NAME 'cim23Location'
     DESC 'The Location class specifies the position and address of a
         PhysicalElement.'
     SUP cim23ManagedElement
     MAY ( cimName $ cimPhysicalPosition $ cimAddress )
   )

   ( <oid-nf14> NAME 'cim23LocationNameForm'
     OC cim23Location
     MUST ( orderedCimKeys )
   )

   ( <sr14> NAME 'cim23LocationStructureRule'
     FORM cim23LocationNameForm
   )
```

The following content rule specifies the auxiliary classes that may be attached to cim23Location.

```
   ( <oid-oc49> NAME 'cim23LocationContentRule'
     DESC 'The auxiliary classes that may be attached to cim23Location'
     AUX ( cim23PhysicalElementLocationAuxClass $
          cim22DependencyAuxClass $ cim23MemberOfCollectionAuxClass )
   )
```

3.2 cim23PhysicalElementLocationAuxClass

This class associates a physical element with a cim23Location object.

```
   ( <oid-at85> NAME 'cimPhysicalLocationRef'
     DESC 'The PhysicalElement"s Location.'
     SYNTAX DN
   )

   ( <oid-oc50> NAME 'cim23PhysicalElementLocationAuxClass'
     DESC 'PhysicalElementLocation associates a PhysicalElement with a
         Location object for inventory or replacement
         purposes. Attribute cimElementRef points to
        cim23PhysicalElement and attribute cimPhysicalLocationRef points
        to cim23Location. '
     SUP top AUXILIARY
     MAY ( cimElementRef $ cimPhysicalLocationRef )
   )
```

3.3 cim23PhysicalCapacity

This class describes a physical element's requirements.

```
( <oid-oc51> NAME 'cim23PhysicalCapacity'
  DESC 'PhysicalCapacity is an abstract class describing a
       PhysicalElement"s minimum/maximum requirements and
       ability to support different types of hardware. For
       example, minimum and maximum memory requirements can be
       modeled as a subclass of cim23PhysicalCapacity.'
  SUP cim23ManagedElement ABSTRACT
  MAY ( cimName )
)
```

3.4 cim23ElementCapacityAuxClass

This class associates a cim23PhysicalCapacity object with one or more cim23PhysicalElements.

```
( <oid-at86> NAME 'cimCapacityRef'
  DESC 'PhysicalCapacity describes the minimum and maximum
      requirements, and ability to support different types of
      hardware for a PhysicalElement.'
  SYNTAX DN
)

( <oid-oc52> NAME 'cim23ElementCapacityAuxClass'
  DESC 'ElementCapacity associates a PhysicalCapacity object with
      one or more PhysicalElements. It serves to associate a
      description of min/max hardware requirements or
      capabilities (stored as a kind of PhysicalCapacity), with
      the PhysicalElements being described. Attribute
      cimCapacityRef points to cim23PhysicalCapacity. Attribute
      cimElementRef points to cim23PhysicalElement.'
  SUP top AUXILIARY
  MAY ( cimCapacityRef $ cimElementRef )
)
```

3.5 cim23MemoryCapacity

Physical elements are limited in what memory can be installed. Instances of this class store information on what memory is currently installed.

```
( <oid-at87> NAME 'cimMemoryType'
  DESC 'The type of memory. This is a part of the object
      key. Values correspond to the list of possible memory types
      in the PhysicalMemory class.   Values are 0="Unknown",
      1="Other", 2="DRAM", 3="Synchronous DRAM", 4="Cache DRAM",
      5="EDO", 6="EDRAM", 7="VRAM", 8="SRAM", 9="RAM", 10="ROM",
      11="Flash", 12="EEPROM", 13="FEPROM", 14="EPROM", 15="CDRAM",
      16="3DRAM", 17="SDRAM", 18="SGRAM", 19="RDRAM".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at88> NAME 'cimMinimumMemoryCapacity'
  DESC 'Minimum amount of memory, in Kbytes, that is needed for
      the associated PhysicalElement to operate correctly.
  SYNTAX integer SINGLE-VALUE
)

( <oid-at89> NAME 'cimMaximumMemoryCapacity'
  DESC 'Maximum amount of memory, in Kbytes, that can be supported
      by the associated PhysicalElement.'
  SYNTAX integer SINGLE-VALUE
```

```
)

( <oid-oc53> NAME 'cim23MemoryCapacity'
  DESC 'MemoryCapacity describes the type of Memory that can be
      installed on a PhysicalElement and its minimum/maximum
      configurations. Information on what memory is currently
      "installed", versus an Element"s min/max requirements, is
      located in instances of the PhysicalMemory class.'
  SUP cim23PhysicalCapacity
  MAY ( cimMemoryType $ cimMinimumMemoryCapacity $
     cimMaximumMemoryCapacity )
)

( <oid-nf16> NAME 'cim23MemoryCapacityNameForm'
  OC cim23MemoryCapacity
  MUST ( orderedCimKeys )
)

( <sr16> NAME 'cim23MemoryCapacityStructureRule'
  FORM cim23MemoryCapacityNameForm
)
```

3.6 cim23ConfigurationCapacity

Capacity includes the number of power supplies, fans, disk drives, etc. that can be connected to or placed on/into a physical element (and the number that must be connected/added/removed at a time). cim23ElementCapacityAuxClass identifies the physical element whose configuration is described.

This class does NOT represent the tradeoffs required of one resource for another. It simply represents capacities. For a StorageLibrary, there are only 2 valid configurations - 9 TapeDrives with 88 Slots, or 3 TapeDrives with 264 Slots. It only conveys that 'up to' 9 Drives and 'up to' 264 slots are available and supported.

```
( <oid-at90> NAME 'cimObjectType'
  DESC 'The type of object (power supply, fan, disk drive, ...)
      whose capacities are indicated. This information is part of
      the class" key.  Values are 0="Other", 1="Processors",
      2="Power Supplies", 3="Fans", 4="Batteries", 5="I/O Slots",
      6="Memory Slots", 7="MediaAccessDevices (Drives)",
      8="StorageMediaLocation Slots", 9="StorageMediaLocation
      Magazines", 10="StorageMediaLocation Panels",
      11="StorageMediaLocation InterLibrary Ports",
      12="StorageMediaLocation Limited Access Ports", 13="Doors",
      14="MediaTransferDevice Pickers", 15="MediaTransferDevice
      Changers", 16="LabelReaders", 17="Contained Chassis",
      18="Connected Chassis", 19="Connected Frames".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at91> NAME 'cimOtherTypeDescription'
  DESC 'A string describing the object type - used when the
      ObjectType property is set to 0 "Other"). OtherTypeDescription
      should be set to NULL when ObjectType is any value other than
      0.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( <oid-at92> NAME 'cimMinimumCapacity'
  DESC 'Minimum number of Elements of type, ObjectType, that must
      be installed.'
```

```
  SYNTAX integer SINGLE-VALUE
)

( <oid-at93> NAME 'cimMaximumCapacity'
  DESC 'Maximum number of Elements of type, ObjectType, that may be
      installed.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at94> NAME 'cimIncrement'
  DESC 'Increment in which Elements must be added or removed.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc54> NAME 'cim23ConfigurationCapacity'
  DESC 'ConfigurationCapacity provides information on the minimum
      and maximum numbers of power supplies, fans, disk drives,
      etc. that can be connected to or placed on/into a
      PhysicalElement (and the number that must be
      connected/added/removed at a time). The PhysicalElement
      whose configuration is described is identified using the
      ElementCapacity association, inherited from
      PhysicalCapacity. The object whose capacities are indicated
      (i.e., the power supply or fan) is identified in the
      ObjectType property of this class. Since the same min/max
      configurations can apply to multiple instances, this class
      is not defined as "weak". Examples of the use of the
      ConfigurationCapacity class are to describe that a "control
      unit" Chassis may be connected to (at most) 4 other I/O
      chassis, or to describe what a StorageLibrary"s cabinet may
      contain. Continuing the latter example, a particular
      StorageLibrary"s cabinet might hold a minimum of 3 and a
      maximum of 9 TapeDrives, and a minimum of 88 and a maximum
      of 264 StorageMediaLocations ("Slots"). This information
      would be described in two instances of
      ConfigurationCapacity, both associated to the
      StorageLibrary"s PhysicalPackage.  This class does NOT
      represent the tradeoffs that are likely to be required of
      one resource for another. It simply represents
      capacities. In the case of the StorageLibrary, there may be
      only 2 valid configurations - 9 TapeDrives with 88 Slots,
      or 3 TapeDrives with 264 Slots. This class only conveys
      that "up to" 9 Drives and "up to" 264 slots may be
      available and are supported.'
  SUP cim23PhysicalCapacity
  MAY ( cimObjectType $ cimOtherTypeDescription $
        cimMinimumCapacity $ cimMaximumCapacity $ cimIncrement )
)

( <oid-nf18> NAME 'cim23ConfigurationCapacityNameForm'
  OC cim23ConfigurationCapacity
  MUST ( orderedCimKeys )
)

( <sr18> NAME 'cim23ConfigurationCapacityStructureRule'
  FORM cim23ConfigurationCapacityNameForm
)
```

3.7 cim23ReplacementSet

A replacement set is a group of physical elements that must be replaced or FRUed together. For example, when replacing a memory card, the component memory chips could be removed and replaced as well.

```
( <oid-oc55> NAME 'cim23ReplacementSet'
  DESC 'The ReplacementSet class aggregates PhysicalElements that
      must be "replaced" or "FRUed" together. For example, when
      replacing a memory card, the component memory chips could
      be removed and replaced as well. Or, a set of memory chips
      may be specified to be replaced or upgraded together using
      this association.'
  SUP top
  MAY ( cimName )
)

( <oid-nf20> NAME 'cim23ReplacementSetNameForm'
  OC cim23ReplacementSet
  MUST ( orderedCimKeys )
)

( <sr20> NAME 'cim23ReplacementSetStructureRule'
  FORM cim23ReplacementSetNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23ReplacementSet.

```
( <oid-oc55> NAME 'cim23ReplacementSetContentRule'
  DESC 'The auxiliary classes that may be attached to
      cim23ReplacementSet'
  AUX ( cim23ParticipatesInSetAuxClass $ cim22DependencyAuxClass $
      cim23MemberOfCollectionAuxClass )
)
```

3.8 cim23ParticipatesInSetAuxClass

This class shows which physical elements should be replaced together.

```
( <oid-at95> NAME 'cimSetRef'
  DESC 'The ReplacementSet.'
  SYNTAX DN
)

( <oid-oc56> NAME 'cim23ParticipatesInSetAuxClass'
  DESC 'ParticipatesInSet indicates which PhysicalElements should
      be replaced together. Attribute cimSetRef points to
      cim23ReplacementSet and attribute cimElementRef points to
      cim23PhysicalElement.'
  SUP top AUXILIARY
  MAY ( cimSetRef $ cimElementRef )
)
```

3.9 cim23PhysicalPackage

A physical package contains or hosts components. Examples are a rack enclosure or an adapter Card.

```
( <oid-at96> NAME 'cimRemovable'
  DESC 'A PhysicalPackage is Removable if it is designed to be
      taken in and out of the physical container in which it is
      normally found, without impairing the function of the
      overall packaging. A Package can still be Removable if
      power must be "off" in order to perform the removal. If
      power can be "on" and the Package removed, then the Element
```

```
      is both Removable and HotSwappable. For example, an extra
      battery in a laptop is Removable, as is a disk drive
      Package inserted using SCA connectors. However, the latter
      is also HotSwappable. A laptop"s display is not Removable,
      nor is a non-redundant power supply. Removing these
      components would impact the function of the overall
      packaging or is impossible due to the tight integration of
      the Package.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at97> NAME 'cimReplaceable'
  DESC 'A PhysicalPackage is Replaceable if it is possible to
      replace (FRU or upgrade) the Element with a physically
      different one. For example, some ComputerSystems allow the
      main Processor chip to be upgraded to one of a higher clock
      rating. In this case, the Processor is said to be
      Replaceable. Another example is a power supply Package
      mounted on sliding rails. All Removable packages are
      inherently Replaceable.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at98> NAME 'cimHotSwappable'
  DESC 'A PhysicalPackage is HotSwappable if it is possible to
      replace the Element with a physically different but
      equivalent one while the containing Package has power
      applied to it (i.e., is "on"). For example, a disk drive
      Package inserted using SCA connectors is both Removable and
      HotSwappable. All HotSwappable packages are inherently
      Removable and Replaceable.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at99> NAME 'cimHeight'
  DESC 'The height of the PhysicalPackage in inches.'
  SUP cimFloat32 SINGLE-VALUE
)

( <oid-at100> NAME 'cimDepth'
  DESC 'The depth of the PhysicalPackage in inches.'
  SUP cimFloat32 SINGLE-VALUE
)

( <oid-at101> NAME 'cimWidth'
  DESC 'The width of the PhysicalPackage in inches.'
  SUP cimFloat32 SINGLE-VALUE
)

( <oid-at102> NAME 'cimWeight'
  DESC 'The weight of the PhysicalPackage in pounds.'
  SUP cimFloat32 SINGLE-VALUE
)

( <oid-oc57> NAME 'cim23PhysicalPackage'
  DESC 'The PhysicalPackage class represents PhysicalElements that
      contain or host other components. Examples are a Rack
      enclosure or an adapter Card.'
  SUP cim23PhysicalElement
  MAY ( cimRemovable $ cimReplaceable $ cimHotSwappable $
     cimHeight $ cimDepth $ cimWidth $ cimWeight )
)

( <oid-nf22> NAME 'cim23PhysicalPackageNameForm'
```

```
  OC cim23PhysicalPackage
  MUST ( orderedCimKeys )
)

( <sr22> NAME 'cim23PhysicalPackageStructureRule'
  FORM cim23PhysicalPackageNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23PhysicalPackage.

```
( <oid-oc57> NAME 'cim23PhysicalPackageContentRule'
  DESC 'The auxiliary classes that may be attached to
      cim23PhysicalPackage'
  AUX ( cim23ContainerAuxClass $ cim23PackageInChassisAuxClass $
     cim23PackagedComponentAuxClass $
     cim23PackageInConnectorAuxClass $
     cim23PackageInSlotAuxClass $ cim23ConnectorOnPackageAuxClass $
     cim22RealizesAuxClass $ cim22ProductPhysicalElementsAuxClass $
     cim22FRUPhysicalElementsAuxClass $
     cim23PhysicalElementLocationAuxClass $
     cim23ElementCapacityAuxClass $ cim23ParticipatesInSetAuxClass $
     cim23ElementsLinkedAuxClass $
     cim22ElementConfigurationAuxClass $
     cim22ElementSettingAuxClass $ cim23CollectedMSEsAuxClass $
     cim22ProvidesServiceToElementAuxClass $ cim22ComponentAuxClass $
     cim22SystemComponentAuxClass $ cim22DependencyAuxClass $
     cim23MemberOfCollectionAuxClass )
)
```

3.10 cim23ContainerAuxClass

This class represents the relationship between a contained and a containing PhysicalElement.

```
( <oid-at103> NAME 'cimLocationWithinContainer'
  DESC 'A free-form string representing the positioning of the
      PhysicalElement within the PhysicalPackage. This string
      could supplement or be used in place of instantiating the
      cimLocation object.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-oc58> NAME 'cim23ContainerAuxClass'
  DESC 'The Container association represents the relationship
      between a contained and a containing PhysicalElement. A
      containing object must be a PhysicalPackage. Attribute
      cimGroupComponentRef points to cim23PhysicalPackage and
      attribute cimPartComponentRef points to
      cim23PhysicalElement.'
  SUP cim22ComponentAuxClass AUXILIARY
  MAY ( cimLocationWithinContainer )
)
```

3.11 cim23PhysicalFrame

A physical frame is a generic frame enclosure.

```
( <oid-at104> NAME 'cimCableManagementStrategy'
  DESC 'CableManagementStrategy is a free-form string that contains
      information on how the various cables are connected and
      bundled for the Frame. With many networking,
```

```
      storage-related and power cables, cable management can be a
      complex and challenging endeavor. This string property
      contains information to aid in assembly and service of the
      Frame.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at105> NAME 'cimLockPresent'
  DESC 'Boolean indicating whether the Frame is protected with a
      lock.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at106> NAME 'cimAudibleAlarm'
  DESC 'Boolean indicating whether the Frame is equipped with an
      audible alarm.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at107> NAME 'cimVisibleAlarm'
  DESC 'Boolean indicating that the equipment includes a visible
      alarm.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at108> NAME 'cimSecurityBreach'
  DESC 'SecurityBreach is an enumerated, integer-valued property
      indicating whether a physical breach of the Frame was
      attempted but unsuccessful (value=4) or attempted and
      successful (5). Also, the values, "Unknown", "Other" or
      "No Breach", can be specified. Values are 1="Other",
      2="Unknown", 3="No Breach", 4="Breach Attempted",
      5="Breach Successful"'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at109> NAME 'cimBreachDescription'
  DESC 'BreachDescription is a free-form string providing more
      information if the SecurityBreach property indicates that a
      breach or some other security-related event occurred.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at110> NAME 'cimIsLocked'
  DESC 'Boolean indicating that the Frame is currently locked.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-oc59> NAME 'cim23PhysicalFrame'
  DESC 'PhysicalFrame is a superclass of Rack, Chassis and other
      frame enclosures, as they are defined in extension classes.
      Properties like visible or audible alarm, and data related to
      security breaches are in this superclass.'
  SUP cim23PhysicalPackage
  MAY ( cimCableManagementStrategy $ cimLockPresent $
     cimAudibleAlarm $ cimVisibleAlarm $ cimSecurityBreach $
     cimBreachDescription $ cimIsLocked )
)

( <oid-nf24> NAME 'cim23PhysicalFrameNameForm'
  OC cim23PhysicalFrame
  MUST ( orderedCimKeys )
)
```

```
( <sr24> NAME 'cim23PhysicalFrameStructureRule'
  FORM cim23PhysicalFrameNameForm
)
```

3.12 cim23Rack

Racks are enclosures in which chassis are placed. Typically they are nothing more than the enclosure, and the chassis packages all functioning components.

```
( <oid-at111> NAME 'cimTypeOfRack'
  DESC 'Enumeration indicating the type of Rack.  Information such
      as "Telco" rack (value=2) or standard 19 inch rack (1) can
      be specified. The country for which the Rack is
      manufactured is defined in the c property.  Values are
      "Unknown", "Standard 19 Inch", "Telco", "Equipment Shelf",
      "Non-Standard".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc60> NAME 'cim23Rack'
  DESC 'A Rack is a PhysicalFrame that represents an enclosure in
      which Chassis are placed. Typically a Rack is nothing more
      than the enclosure, and all the functioning componentry is
      packaged in the Chassis, loaded in the Rack.'
  SUP cim23PhysicalFrame
  MAY ( cimTypeOfRack $ c )
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23Rack.

```
( <oid-oc60> NAME 'cim23RackContentRule'
  DESC 'The auxiliary classes that may be attached to cim23Rack'
  AUX ( cim23ChassisInRackAuxClass )
)
```

3.13 cim23Chassis

Chassis enclose other elements and provide definable functionality.

```
( <oid-at112> NAME 'cimNumberOfPowerCords'
  DESC 'Integer indicating the number of power cords which must be
      connected to the Chassis, for all the componentry to
      operate.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at113> NAME 'cimCurrentRequiredOrProduced'
  DESC 'Current required by the Chassis at 120V. If power is
      provided by the Chassis (as in the case of a UPS), this
      property may indicate the amperage produced, as a negative
      number.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at114> NAME 'cimHeatGeneration'
  DESC 'Amount of heat generated by the Chassis in BTU/hour.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc61> NAME 'cim23Chassis'
  DESC 'The Chassis class represents the PhysicalElements that
```

```
      enclose other Elements and provide definable functionality,
      such as a desktop, processing node, UPS, disk or tape
      storage, or a combination of these.'
  SUP cim23PhysicalFrame
  MAY ( cimNumberOfPowerCords $ cimCurrentRequiredOrProduced $
     cimHeatGeneration )
)

( <oid-nf26> NAME 'cim23ChassisNameForm'
  OC cim23Chassis
  MUST ( orderedCimKeys )
)

( <sr26> NAME 'cim23ChassisStructureRule'
  FORM cim23ChassisNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23Chassis.

```
( <oid-oc61> NAME 'cim23ChassisContentRule'
  DESC 'The auxiliary classes that may be attached to cim23Chassis'
  AUX ( cim23ChassisInRackAuxClass $ cim23PackageInChassisAuxClass $
     cim23DockedAuxClass )
)
```

3.14 cim23ChassisInRackAuxClass

This class makes explicit the 'containing' relationship between the Rack and the Chassis.

```
( <oid-at115> NAME 'cimBottomU'
  DESC 'An integer indicating the lowest or "bottom" U in which the
      Chassis is mounted. A "U" is a standard unit of measure for
      the height of a Rack or rack-mountable component. It is
      equal to 1.75 inches or 4.445 cm.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc62> NAME 'cim23ChassisInRackAuxClass'
  DESC 'Racks, as simple enclosures, contain Chassis that provide
      the physical componentry realizing processing nodes,
      storage devices, UPSs, etc. The ChassisInRack association
      makes explicit the "containing" relationship between the
      Rack and the Chassis.  Attribute cimGroupComponentRef points
      to cim23Rack and attribute cimPartComponentRef points to
      cim23Chassis.'
  SUP cim23ContainerAuxClass AUXILIARY
  MAY ( cimLocationWithinContainer $ cimBottomU )
)
```

3.15 cim23PackageInChassisAuxClass

This class makes the containment relationship between a chassis and other packages explicit.

```
( <oid-oc63> NAME 'cim23PackageInChassisAuxClass'
  DESC 'A Chassis can contain other Packages, such as other Chassis
      and Cards. The PackageInChassis association makes explicit
      this relationship. Attribute cimGroupComponentRef points to
      cim23Chassis and attribute cimPartComponentRef points to
      cim23PhysicalPackage.'
  SUP cim23ContainerAuxClass AUXILIARY
```

```
    MAY ( cimLocationWithinContainer )
  )
```

3.16 cim23DockedAuxClass

This class makes explicit the relationship between a laptop, a type of chassis, which docks in another type of chassis, a docking station.

```
  ( <oid-oc64> NAME 'cim23DockedAuxClass'
    DESC 'A laptop, a type of Chassis, may be docked in another type
        of Chassis, a Docking Station. This is the relationship
        represented by the Docked association. Because this is such
        a typical relationship, it is explicitly described. Both
        attributes point to cim23Chassis objects.'
    SUP cim22DependencyAuxClass AUXILIARY
)
```

3.17 cim23Card

This class represents a type of physical container that can be plugged into another Card or HostingBoard, or is itself a HostingBoard/Motherboard in a Chassis. It includes any package capable of carrying signals and providing a mounting point for PhysicalComponents, such as Chips, or other PhysicalPackages, such as other Cards.

```
  ( <oid-at116> NAME 'cimHostingBoard'
    DESC 'Boolean indicating that this Card is a Motherboard or, more
        generically, a baseboard in a Chassis.'
    SYNTAX boolean SINGLE-VALUE
  )

  ( <oid-at117> NAME 'cimSlotLayout'
    DESC 'SlotLayout is a free-form string that describes the slot
        positioning, typical usage, restrictions, individual slot
        spacings or any other pertinent information for the slots
        on a Card.'
    SYNTAX DirectoryString SINGLE-VALUE
  )

  ( <oid-at118> NAME 'cimRequiresDaughterBoard'
    DESC 'Boolean indicating that at least one daughterboard or
        auxiliary Card is required in order to function properly.'
    SYNTAX boolean SINGLE-VALUE
  )

  ( <oid-at119> NAME 'cimSpecialRequirements'
    DESC 'Boolean indicating that this Card is physically unique
        from other Cards of the same type and therefore requires a
        special Slot. For example, a doublewide Card requires two
        Slots. Another example is where a certain Card may be used
        for the same general function as other Cards but requires a
        special Slot (e.g., extra long), whereas the other Cards
        can be placed in any available Slot. If set to TRUE, then
        the corresponding property, RequirementsDescription, should
        specify the nature of the uniqueness or purpose of the
        Card.'
    SYNTAX boolean SINGLE-VALUE
  )

  ( <oid-at120> NAME 'cimRequirementsDescription'
    DESC 'A free-form string describing the way(s) in which this Card
        is physically unique from other Cards. This property only
        has meaning when the corresponding boolean property,
```

```
      SpecialRequirements, is set to TRUE.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at121> NAME 'cimOperatingVoltages'
  DESC 'Operating voltages required by the Card.'
  SYNTAX integer
)

( <oid-oc65> NAME 'cim23Card'
  DESC 'The Card class represents a type of physical container that
      can be plugged into another Card or HostingBoard, or is
      itself a HostingBoard/Motherboard in a Chassis. The
      cim23Card class includes any package capable of carrying
      signals and providing a mounting point for
      PhysicalComponents, such as Chips, or other
      PhysicalPackages, such as other Cards.'
  SUP cim23PhysicalPackage
  MAY ( cimHostingBoard $ cimSlotLayout $ cimRequiresDaughterBoard $
     cimSpecialRequirements $ cimRequirementsDescription $
     cimOperatingVoltages )
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23Card.

```
( <oid-oc65> NAME 'cim23CardContentRule'
  DESC 'The auxiliary classes that may be attached to cim23Card'
  AUX ( cim23CardOnCardAuxClass $ cim23MemoryOnCardAuxClass $
     cim23CardInSlotAuxClass )
)
```

3.18 cim23SystemBusCard

System bus cards require additional attributes, detailing the card's bus type and data width, which dictate the type of slot into which the card can be inserted. For example, attributes can define that a card is a PCI 64-bit adapter.

```
( <oid-at122> NAME 'cimBusType'
  DESC 'An enumerated integer describing the System bus type for
      this Card. It indicates the type of Slot into which the
      Card can plug. Values: 43="PCI", 44="ISA", 45="EISA",
      46="VESA", 47="PCMCIA", 48="PCMCIA Type I", 49="PCMCIA Type
      II", 50="PCMCIA Type III", 52="CardBus", 64="Access.bus",
      65="NuBus", 73="AGP", 74="VME Bus", 75="VME64",
      76="Proprietary", 77="Proprietary Processor Card Slot",
      78="Proprietary Memory Card Slot", 79="Proprietary I/O
      Riser Slot", 80="PCI-66MHZ", 81="AGP2X", 82="AGP4X",
      83="PC-98", 84="PC-98-Hireso", 85="PC-H98", 86="PC-98Note",
      87="PC-98Full", 98="PCI-X", 99="Sbus IEEE 1396-1993 32
      bit", 100="Sbus IEEE 1396-1993 64 bit", 101="MCA",
      102="GIO", 103="XIO", 104="HIO", 105="NGIO", 106="PMC",
      109="Future I/O", 110="InfiniBand"'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at123> NAME 'cimBusWidth'
  DESC 'System bus width (in bits) required by this Card. If
      "unknown", enter 0. If "other" than the values, 8, 16, 32,
      64 or 128, enter 1. Values are 0, 1, 8, 16, 32, 64, and 128'
  SYNTAX integer SINGLE-VALUE
)
```

```
( <oid-oc66> NAME 'cim23SystemBusCard'
  DESC 'The SystemBusCard class represents additional information
      for a cimCard, detailing the Card"s bus type and data
      width. These properties dictate the type of Slot into which
      the Card can be inserted. For example, using the properties
      of this class, one can define that a Card is a PCI, 64 bit
      adapter.'
  SUP cim23Card
  MAY ( cimBusType $ cimBusWidth )
)
```

3.19 cim23CardOnCardAuxClass

Cards may be plugged into Motherboards/baseboards, are daughtercards of an adapter, or support special Card-like modules. This auxiliary class describes these relationships.

```
( <oid-at124> NAME 'cimMountOrSlotDescription'
  DESC 'A string describing and identifying how the Card is mounted
      on or plugged into the "other" Card. Slot information could
      be included in this field and may be sufficient for certain
      management purposes. If so, this avoids creating
      instantiations of Connector/Slot objects just to model the
      relationship of Cards to HostingBoards or other
      adapters. On the other hand, if Slot and Connector
      information is available, this field could be used to
      provide more detailed mounting or slot insertion data.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-oc67> NAME 'cim23CardOnCardAuxClass'
  DESC 'Cards may be plugged into Motherboards/baseboards, are
      daughtercards of an adapter, or support special Card-like
      modules. These relationships are described by the
      CardOnCard association. Both reference attributes point to
      cim23Card objects.'
  SUP cim23ContainerAuxClass AUXILIARY
  MAY ( cimLocationWithinContainer $ cimMountOrSlotDescription )
)
```

3.20 cim23StorageMediaLocation

A storage media location holds media and goes beyond being just a location used by a storage library.

```
( <oid-at125> NAME 'cimLocationType'
  DESC 'The type of Location. For example, whether this is an
      individual Media "Slot" (value=2), a MediaAccessDevice
      (value=4) or a "Magazine" (value=3) is indicated in this
      property.  Values are 0="Unknown", 1="Other", 2="Slot",
      3="Magazine", 4="MediaAccessDevice", 5="InterLibrary Port",
      6="Limited Access Port", 7="Door", 8="Shelf", 9="Vault".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at126> NAME 'cimLocationCoordinates'
  DESC 'LocationCoordinates represent the physical location of the
      the StorageMediaLocation instance. The property is defined
      as a free-form string to allow the location information to
      be described in vendor-unique terminology.'
  SYNTAX DirectoryString SINGLE-VALUE
)
```

```
( <oid-at127> NAME 'cimMediaCapacity'
  DESC 'A StorageMediaLocation may hold more than one PhysicalMedia
      - for example, a Magazine. This property indicates the
      PhysicalMedia capacity of the Location.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc68> NAME 'cim23StorageMediaLocation'
  DESC 'StorageMediaLocation is a PhysicalElement where
      PhysicalMedia may be placed. This class describes an entity
      that holds Media and is not just a "place" (as is conveyed
      by the cim23Location object). This class is typically used
      in the context of a StorageLibrary. Examples of
      StorageMediaLocations are MediaAccessDevices,
      InterLibraryPorts or "slots" in a Library panel.'
  SUP cim23PhysicalPackage
  MAY ( cimLocationType $ cimLocationCoordinates $ cimMediaCapacity )
)

( <oid-nf28> NAME 'cim23StorageMediaLocationNameForm'
  OC cim23StorageMediaLocation
  MUST ( orderedCimKeys )
)

( <sr28> NAME 'cim23StorageMediaLocationStructureRule'
  FORM cim23StorageMediaLocationNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23StorageMediaLocation.

```
( <oid-oc68> NAME 'cim23StorageMediaLocationContentRule'
  DESC 'The auxiliary classes that may be attached to
      cim23StorageMediaLocation'
  AUX ( cim23PhysicalMediaInLocationAuxClass )
)
```

3.21 cim23PhysicalComponent

A physical component either can not or does not need to be decomposed into its constituent parts. For example, an ASIC can not be further decomposed and a tape for data storage does not need to be decomposed. Any element that is not a link, connector, or package is subclassed from this class.

```
( <oid-oc69> NAME 'cim23PhysicalComponent'
  DESC 'The PhysicalComponent class represents any low-level or
      basic Component within a Package. A Component object either
      can not or does not need to be decomposed into its
      constituent parts. For example, an ASIC (or Chip) can not
      be further decomposed. A tape for data storage
      (PhysicalMedia) does not need to be decomposed. Any
      PhysicalElement that is not a Link, Connector, or Package
      is a descendent (or member) of this class. For example, the
      UART chipset on an internal modem Card would be a subclass
      (if additional properties or associations are defined) or
      an instance of PhysicalComponent.'
  SUP cim23PhysicalElement
  MAY ( cimRemovable $ cimReplaceable $ cimHotSwappable )
)

( <oid-nf30> NAME 'cim23PhysicalComponentNameForm'
```

```
  OC cim23PhysicalComponent
  MUST ( orderedCimKeys )
)

( <sr30> NAME 'cim23PhysicalComponentStructureRule'
  FORM cim23PhysicalComponentNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23PhysicalComponent.

```
( <oid-oc69> NAME 'cim23PhysicalComponentContentRule'
  DESC 'The auxiliary classes that may be attached to
      cim23PhysicalComponent'
  AUX ( cim23PackagedComponentAuxClass $ cim22RealizesAuxClass $
     cim22ProductPhysicalElementsAuxClass $
     cim22FRUPhysicalElementsAuxClass $
     cim23PhysicalElementLocationAuxClass $
     cim23ElementCapacityAuxClass $
     cim23ParticipatesInSetAuxClass $ cim23ContainerAuxClass $
     cim23ElementsLinkedAuxClass $
     cim22ElementConfigurationAuxClass $
     cim22ElementSettingAuxClass $ cim23CollectedMSEsAuxClass $
     cim22ProvidesServiceToElementAuxClass $ cim22ComponentAuxClass $
     cim22SystemComponentAuxClass $ cim22DependencyAuxClass $
     cim23MemberOfCollectionAuxClass )
)
```

3.22 cim23PackagedComponentAuxClass

As a physical package typically contains a component, this class makes this relationship explicit. The word, 'typically', is used because a Component may be removed from, or not yet inserted into, its containing Package (i.e., the Removable boolean is TRUE). Therefore, a Component may not always be associated with a container.

```
( <oid-oc70> NAME 'cim23PackagedComponentAuxClass'
  DESC 'A Component is typically contained by a PhysicalPackage,
      such as a Chassis or Card. The PackagedComponent
      association makes this relationship explicit. In the first
      sentence, the word, "typically", is used. This is because a
      Component may be removed from, or not yet inserted into,
      its containing Package (i.e., the Removable boolean is
      TRUE). Therefore, a Component may not always be associated
      with a container. Attribute cimGroupComponentRef points to
      cim23PhysicalPackage and attribute cimPartComponentRef points
      to cim23PhysicalComponent.'
  SUP cim23ContainerAuxClass AUXILIARY
  MAY ( cimLocationWithinContainer )
)
```

3.23 cim23Chip

A chip is of IC hardware, including ASICs, processors, and memory chips.

```
( <oid-at128> NAME 'cimFormFactor'
  DESC 'The implementation form factor for the Chip.  For example,
      values such as SIMM (7), TSOP (9) or PGA (10) can be
      specified. Values are 0="Unknown", 1="Other", 2="SIP",
      3="DIP", 4="ZIP", 5="SOJ", 6="Proprietary", 7="SIMM",
      8="DIMM", 9="TSOP", 10="PGA", 11="RIMM", 12="SODIMM",
      13="SRIMM", 14="SMD", 15="SSMP", 16="QFP", 17="TQFP",
```

```
      18="SOIC", 19="LCC", 20="PLCC", 21="BGA", 22="FPBGA",
      23="LGA".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-oc71> NAME 'cim23Chip'
  DESC 'The Chip class represents any type of integrated circuit
      hardware, including ASICs, processors, memory chips, etc.'
  SUP cim23PhysicalComponent
  MAY ( cimFormFactor )
)
```

3.24 cim23PhysicalMemory

Low level memory devices are examples of physical memory.

```
( <oid-at129> NAME 'cimTotalWidth'
  DESC 'Total width, in bits, of the PhysicalMemory, including
      check or error correction bits. If there are no error
      correction bits, the value in this property should match
      that specified for DataWidth.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at130> NAME 'cimDataWidth'
  DESC 'Data width of the PhysicalMemory, in bits. A data width of
      0 and a TotalWidth of 8 would indicate that the Memory is
      solely used to provide error correction bits.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at131> NAME 'cimSpeed'
  DESC 'The speed of the PhysicalMemory, in nanoseconds.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at132> NAME 'cimCapacity'
  DESC 'The total capacity of this PhysicalMemory, in bytes.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at133> NAME 'cimBankLabel'
  DESC 'A string identifying the physically labeled bank where the
      Memory is located - for example, "Bank 0" or "Bank A".'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( <oid-at134> NAME 'cimPositionInRow'
  DESC 'Specifies the position of the PhysicalMemory in a
      "row". For example, if it takes two 8-bit memory devices to
      form a 16-bit row, then a value of "2"means that this
      Memory is the second device. 0 is an invalid value for this
      property.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at135> NAME 'cimInterleavePosition'
  DESC 'The position of this PhysicalMemory in an interleave. 0
      indicates non-interleaved. 1 indicates the first position,
      2 the second position and so on. For example, in a 2:1
      interleave, a value of "1" would indicate that the Memory
      is in the "even" position.'
  SYNTAX integer SINGLE-VALUE
)
```

```
( <oid-oc72> NAME 'cim23PhysicalMemory'
  DESC 'PhysicalMemory is a subclass of cim23Chip, representing low
      level memory devices - SIMMS, DIMMs, raw memory chips,
      etc.'
  SUP cim23Chip
  MAY ( cimFormFactor $ cimMemoryType $ cimTotalWidth $
     cimDataWidth $ cimSpeed $ cimCapacity $ cimBankLabel $
     cimPositionInRow $ cimInterleavePosition )
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23PhysicalMemory.

```
( <oid-oc72> NAME 'cim23PhysicalMemoryContentRule'
  DESC 'The auxiliary classes that may be attached to
      cim23PhysicalMemory'
  AUX ( cim23MemoryOnCardAuxClass $ cim23MemoryWithMediaAuxClass )
)
```

3.25 cim23MemoryOnCardAuxClass

Hosting boards, adapter Cards, etc., can hold physical memory. Therefore, this class represents that relationship.

```
( <oid-oc73> NAME 'cim23MemoryOnCardAuxClass'
  DESC 'PhysicalMemory can be located on HostingBoards, adapter
      Cards, etc. This association explicitly defines this
      relationship of Memory to Cards.  Attribute
      cimGroupComponentRef points to cim23Card. Attribute
      cimPartComponentRef points to cim23PhysicalMemory.'
  SUP cim23PackagedComponentAuxClass AUXILIARY
  MAY ( cimLocationWithinContainer )
)
```

3.26 cim23PhysicalMedia

Physical media are any type of documentation or storage medium, typically removable media. However, this class can also model 'sealed' media where cim23PackagedComponentAuxClass associates the media with the physical package.

```
( <oid-at136> NAME 'cimMediaType'
  DESC 'Specifies the type of the PhysicalMedia, as an enumerated
      integer. The MediaDescription property is used to provide
      more explicit definition of the Media type, whether it is
      pre-formatted, compatability features, etc. Values are
      0="Unknown", 1="Other", 2="Tape Cartridge", 3="QIC
      Cartridge", 4="AIT Cartridge", 5="DTF Cartridge", 6="DAT
      Cartridge", 7="8mm Tape Cartridge", 8="19mm Tape
      Cartridge", 9="DLT Cartridge", 10="Half-Inch Magnetic Tape
      Cartridge", 11="Cartridge Disk", 12="JAZ Disk", 13="ZIP
      Disk", 14="SyQuest Disk", 15="Winchester Removable Disk",
      16="CD-ROM", 17="CD-ROM/XA", 18="CD-I", 19="CD Recordable",
      20="WORM", 21="Magneto-Optical", 22="DVD", 23="DVD-RW+",
      24="DVD-RAM", 25="DVD-ROM", 26="DVD-Video", 27="Divx",
      28="Floppy/Diskette", 29="Hard Disk", 30="Memory Card",
      31="Hard Copy", 32="Clik Disk", 33="CD-RW", 34="CD-DA",
      35="CD+", 36="DVD Recordable", 37="DVD-RW", 38="DVD-Audio",
      39="DVD-5", 40="DVD-9", 41="DVD-10", 42="DVD-18",
      43="Magneto-Optical Rewriteable", 44="Magneto-Optical Write
      Once", 45="Magneto-Optical Rewriteable (LIMDOW)", 46="Phase
```

```
      Change Write Once", 47="Phase Change Rewriteable",
      48="Phase Change Dual Rewriteable", 49="Ablative Write
      Once", 50="Near Field Recording", 51="MiniQic",
      52="Travan", 53="8mm Metal Particle", 54="8mm Advanced
      Metal Evaporate", 55="NCTP", 56="LTO Ultrium", 57="LTO
      Accelis", 58="9 Track Tape", 59="18 Track Tape", 60="36
      Track Tape", 61="Magstar 3590", 62="Magstar MP", 63="D2
      Tape", 64="Tape - DST Small ", 65="Tape - DST Medium",
      66="Tape - DST Large".'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at137> NAME 'cimMediaDescription'
  DESC 'Additional detail related to the MediaType enumeration. For
      example, if value 3 ("QIC Cartridge") is specified, this
      property could indicate whether the tape is wide or 1/4
      inch, whether it is pre-formatted, whether it is Travan
      compatible, etc.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at138> NAME 'cimWriteProtectOn'
  DESC 'Boolean specifying whether the Media is currently write
      protected by some kind of physical mechanism, such as a
      protect tab on a floppy diskette.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at139> NAME 'cimCleanerMedia'
  DESC 'Boolean indicating that the PhysicalMedia is used for
      cleaning purposes and not data storage.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at140> NAME 'cimMediaSize'
  DESC 'Size of the Media in inches. For example, "3.5" would be
      entered for a 3.5 inch disk, or "12" would be entered for a
      12 inch optical disk. On the other hand, "0.5" would be
      defined for a 1/2 inch tape.'
  SUP cim23Float32 SINGLE-VALUE
)

( <oid-at141> NAME 'cimMaxMounts'
  DESC 'For removable Media, the maximum number of times that the
      Media can be mounted before it should be retired. For
      cleaner Media, this is the maximum number of Drive cleans
      that can be performed. For nonremovable Media, such as hard
      disks, this property is not applicable and should be set to
      0.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at142> NAME 'cimDualSided'
  DESC 'Boolean indicating that the Media has two recording sides
      (TRUE) or only a single side (FALSE). Examples of dual
      sided Media include DVD-ROM and some optical
      disks. Examples of single sided Media are tapes and
      CD-ROM.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-oc74> NAME 'cim23PhysicalMedia'
  DESC 'The PhysicalMedia class represents any type of
      documentation or storage medium, such as tapes, CDROMs,
```

```
        etc. This class is typically used to locate and manage
        Removable Media (versus Media sealed with the
        MediaAccessDevice, as a single Package, as is the case with
        hard disks). However, "sealed" Media can also be modeled
        using this class, where the Media would then be associated
        with the PhysicalPackage using the PackagedComponent
        relationship.'
    SUP cim23PhysicalComponent
    MAY ( cimCapacity $ cimMediaType $ cimMediaDescription $
       cimWriteProtectOn $ cimCleanerMedia $ cimMediaSize $
       cimMaxMounts $ cimDualSided $ cimPhysicalLabels $
       cimLabelStates $ cimLabelFormats )
)

( <oid-nf32> NAME 'cim23PhysicalMediaNameForm'
  OC cim23PhysicalMedia
  MUST ( orderedCimKeys )
)

( <sr32> NAME 'cim23PhysicalMediaStructureRule'
  FORM cim23PhysicalMediaNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23PhysicalMedia.

```
( <oid-oc74> NAME 'cim23PhysicalMediaContentRule'
  DESC 'The auxiliary classes that may be attached to
       cim23PhysicalMedia'
  AUX ( cim23MemoryWithMediaAuxClass $
     cim23PhysicalMediaInLocationAuxClass )
)
```

3.27 cim23MemoryWithMediaAuxClass

This class shows that memory is associated with a physical media and its cartridge and provides identification and also stores user-specific data.

```
( <oid-oc75> NAME 'cim23MemoryWithMediaAuxClass'
  DESC 'MemoryWithMedia indicates that Memory is associated with a
       PhysicalMedia and its cartridge. The Memory provides media
       identification and also stores user-specific
       data. Attribute cimAntecedentRef points to
       cim23PhysicalMemory and attribute cimDependentRef points to
       cim23PhysicalMedia.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.28 cim23PhysicalMediaInLocationAuxClass

Within a storage library, all media should be accounted for, and be present in some storage location. In addition, one can determine if a location is empty or full based on whether this auxiliary class is attached to a cim23StorageMediaLocation object.

```
( <oid-oc76> NAME 'cim23PhysicalMediaInLocationAuxClass'
  DESC 'Within a StorageLibrary, all Media should be accounted for,
       and be present in some Storage Location. This relationship
       is made explicit by the PhysicalMediaInLocation
       association.  In addition, one can determine if a Location is
       empty or full based on whether this association exists for
```

```
        the StorageMediaLocation. Attribute cimAntecedentRef points
        to cim23StorageMediaLocation and attribute cimDependentRef
        points to cim23PhysicalMedia.'
    SUP cim22DependencyAuxClass AUXILIARY
)
```

3.29 cim23PhysicalTape

This class represents data for a tape Media, including information on the length and whether it must be unloaded from BOT.

```
( <oid-at143> NAME 'cimTapeLength'
  DESC 'The physical length of the Tape in feet.'
  SUP cim23Float32 SINGLE-VALUE
)

( <oid-at144> NAME 'cimUnloadAnywhere'
  DESC 'Boolean set to TRUE if the Tape can be unloaded at any
      position on the Media. It is set to FALSE if the tape must
      be at a certain position for unload - such as at the
      beginning of tape (BOT) area, or at mid-tape point for
      TapeDrives with mid-tape load.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-oc77> NAME 'cim23PhysicalTape'
  DESC 'The PhysicalTape class represents additional data for a
      Tape Media. Information on the tape length and whether it
      must be unloaded from BOT are properties of this class.'
  SUP cim23PhysicalMedia
  MAY ( cimTapeLength $ cimUnloadAnywhere )
)
```

3.30 cim23PhysicalLink

Physical links are the cabling together of physical elements, including cables and links. Rather than model the numerous physical cables within a physical package or network, this class is intended for those cases where the cables or links are either critical components or important assets.

```
( <oid-at145> NAME 'cimMaxLength'
  DESC 'The maximum length of the PhysicalLink in feet.'
  SUP cim23Float64 SINGLE-VALUE
)

( <oid-at146> NAME 'cimLength'
  DESC 'The current length of the PhysicalLink in feet. For some
      connections, especially wireless technologies, this
      property may not be applicable and should be left
      uninitialized.'
  SUP cim23Float64 SINGLE-VALUE
)

( <oid-at147> NAME 'cimWired'
  DESC 'Boolean indicating whether the PhysicalLink is an actual
      cable (TRUE) or a wireless connection (FALSE).'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-oc78> NAME 'cim23PhysicalLink'
  DESC 'The PhysicalLink class represents the cabling of
      PhysicalElements together. For example, serial or Ethernet
```

```
        cables, and infrared Links would be subclasses (if
        additional properties or associations are defined) or
        instances of PhysicalLink. In many cases, the numerous
        physical cables within a PhysicalPackage or Network will
        not be modeled. However, where these cables or Links are
        critical components, or are tagged assets of the company,
        these objects can be instantiated using this class or one
        of its descendent classes.'
    SUP cim23PhysicalElement
    MAY ( cimMaxLength $ cimLength $ cimWired $ cimMediaType )
)

( <oid-nf34> NAME 'cim23PhysicalLinkNameForm'
    OC cim23PhysicalLink
    MUST ( orderedCimKeys )
)

( <sr34> NAME 'cim23PhysicalLinkStructureRule'
    FORM cim23PhysicalLinkNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23PhysicalLink.

```
( <oid-oc78> NAME 'cim23PhysicalLinkContentRule'
    DESC 'The auxiliary classes that may be attached to
       cim23PhysicalLink'
    AUX ( cim23ElementsLinkedAuxClass $ cim23LinkHasConnectorAuxClass $
          cim22RealizesAuxClass $ cim22ProductPhysicalElementsAuxClass $
          cim22FRUPhysicalElementsAuxClass $
          cim23PhysicalElementLocationAuxClass $
          cim23ElementCapacityAuxClass $
          cim23ParticipatesInSetAuxClass $ cim23ContainerAuxClass $
          cim22ElementConfigurationAuxClass $
          cim22ElementSettingAuxClass $ cim23CollectedMSEsAuxClass $
          cim22ProvidesServiceToElementAuxClass $
          cim22ComponentAuxClass $ cim22SystemComponentAuxClass $
          cim22DependencyAuxClass $ cim23MemberOfCollectionAuxClass )
)
```

3.31 cim23ElementsLinkedAuxClass

This class shows which physical elements are cabled together by a physical link.

```
( <oid-oc79> NAME 'cim23ElementsLinkedAuxClass'
    DESC 'The ElementsLinked association indicates which
        PhysicalElements are cabled together by a
        PhysicalLink. Attribute cimAntecedentRef points to
        cim23PhysicalLink and attribute cimDependentRef points to
        cim23PhysicalElement.'
    SUP cim22DependencyAuxClass AUXILIARY
)
```

3.32 cim23PhysicalConnector

This class represents any physical element that is used to connect to other elements. Any object that can be used to connect and transmit signals or power between two or more physical elements is a descendant of this class. For example, slots and D-shell connectors are types of physical connectors.

```
( <oid-at148> NAME 'cimConnectorPinout'
```

```
  DESC 'A free-form string describing the pin configuration and
      signal usage of a PhysicalConnector.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at149> NAME 'cimConnectorType'
  DESC 'An array of integers defining the type of
      PhysicalConnector. An array is specified to allow the
      description of "combinations" of Connector information. For
      example, one array entry could specify RS-232 (value=25),
      another DB-25 (value=23) and a third entry define the
      Connector as "Male" (value=2). Values are 0="Unknown",
      1="Other", 2="Male", 3="Female", 4="Shielded",
      5="Unshielded", 6="SCSI (A) High-Density (50 pins)",
      7="SCSI (A) Low-Density (50 pins)", 8="SCSI (P)
      High-Density (68 pins)", 9="SCSI SCA-I (80 pins)", 10="SCSI
      SCA-II (80 pins)", 11="Fibre Channel (DB-9, Copper)",
      12="Fibre Channel (Optical Fibre)", 13="Fibre Channel
      SCA-II (40 pins)", 14="Fibre Channel SCA-II (20 pins)",
      15="Fibre Channel BNC", 16="ATA 3-1/2 Inch (40 pins)",
      17="ATA 2-1/2 Inch (44 pins)", 18="ATA-2", 19="ATA-3",
      20="ATA/66", 21="DB-9", 22="DB-15", 23="DB-25", 24="DB-36",
      25="RS-232C", 26="RS-422", 27="RS-423", 28="RS-485",
      29="RS-449", 30="V.35", 31="X.21", 32="IEEE-488", 33="AUI",
      34="UPT Category 3", 35="UPT Category 4", 36="UPT Category
      5", 37="BNC", 38="RJ11", 39="RJ45", 40="Fiber MIC",
      41="Apple AUI", 42="Apple GeoPort", 43="PCI", 44="ISA",
      45="EISA", 46="VESA", 47="PCMCIA", 48="PCMCIA Type I",
      49="PCMCIA Type II", 50="PCMCIA Type III", 51="ZV Port",
      52="CardBus", 53="USB", 54="IEEE 1394", 55="HIPPI",
      56="HSSDC (6 pins)", 57="GBIC", 58="DIN", 59="Mini-DIN",
      60="Micro-DIN", 61="PS/2", 62="Infrared", 63="HP-HIL",
      64="Access.bus", 65="NuBus", 66="Centronics",
      67="Mini-Centronics", 68="Mini-Centronics Type-14",
      69="Mini-Centronics Type-20", 70="Mini-Centronics Type-26",
      71="Bus Mouse", 72="ADB", 73="AGP", 74="VME Bus",
      75="VME64", 76="Proprietary", 77="Proprietary Processor
      Card Slot", 78="Proprietary Memory Card Slot",
      79="Proprietary I/O Riser Slot", 80="PCI-66MHZ",
      81="AGP2X", 82="AGP4X", 83="PC-98", 84="PC-98-Hireso",
      85="PC-H98", 86="PC-98Note", 87="PC-98Full", 88="SSA SCSI",
      89="Circular", 90="On Board IDE Connector", 91="On Board
      Floppy Connector", 92="9 Pin Dual Inline", 93="25 Pin Dual
      Inline", 94="50 Pin Dual Inline", 95="68 Pin Dual Inline",
      96="On Board Sound Connector", 97="Mini-jack", 98="PCI-X",
      99="Sbus IEEE 1396-1993 32 bit", 100="Sbus IEEE 1396-1993
      64 bit", 101="MCA", 102="GIO", 103="XIO", 104="HIO",
      105="NGIO", 106="PMC", 107="MTRJ", 108="VF-45", 109="Future
      I/O", 110="SC", 111="SG", 112="Electrical", 113="Optical",
      114="Ribbon", 115="GLM", 116="1x9", 117="Mini SG",
      118="LC", 119="HSSC", 120="VHDCI Shielded (68 pins)",
      121="InfiniBand".'
  SYNTAX integer
)

( <oid-oc80> NAME 'cim23PhysicalConnector'
  DESC 'The PhysicalConnector class represents any PhysicalElement
      that is used to connect to other Elements. Any object that
      can be used to connect and transmit signals or power
      between two or more PhysicalElements is a descendant (or
      member) of this class. For example, Slots and D-shell
      connectors are types of PhysicalConnectors.'
  SUP cim23PhysicalElement
  MAY ( cimConnectorPinout $ cimConnectorType $
```

```
        cimOtherTypeDescription )
)

( <oid-nf36> NAME 'cim23PhysicalConnectorNameForm'
  OC cim23PhysicalConnector
  MUST ( orderedCimKeys )
)

( <sr36> NAME 'cim23PhysicalConnectorStructureRule'
  FORM cim23PhysicalConnectorNameForm
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23PhysicalConnector.

.sp
.nf

```
 ( <oid-oc80> NAME 'cim23PhysicalConnectorContentRule'
   DESC 'The auxiliary classes that may be attached to
       cim23PhysicalConnector'
   AUX ( cim23ConnectedToAuxClass $ cim23PackageInConnectorAuxClass $
        cim23LinkHasConnectorAuxClass $
        cim23ConnectorOnPackageAuxClass $ cim22RealizesAuxClass $
        cim22ProductPhysicalElementsAuxClass $
        cim22FRUPhysicalElementsAuxClass $
        cim23PhysicalElementLocationAuxClass $
        cim23ElementCapacityAuxClass $
        cim23ParticipatesInSetAuxClass $ cim23ContainerAuxClass $
        cim23ElementsLinkedAuxClass $
        cim22ElementConfigurationAuxClass $
        cim22ElementSettingAuxClass $ cim23CollectedMSEsAuxClass $
        cim22ProvidesServiceToElementAuxClass $
        cim22ComponentAuxClass $ cim22SystemComponentAuxClass $
        cim22DependencyAuxClass $ cim23MemberOfCollectionAuxClass )
)
```

3.33 cim23ConnectedToAuxClass

This class shows that two or more physical connectors are connected.

```
( <oid-oc81> NAME 'cim23ConnectedToAuxClass'
  DESC 'The ConnectedTo association indicates that two or more
      PhysicalConnectors are connected together. Both attributes
      point to cim23PhysicalConnector objects.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.34 cim23Slot

A slot represents connectors into which packages are inserted. For example, a physical package that is a disk drive may be inserted into a SCA slot. As another example, a card may be inserted into a 16-, 32-, or 64-bit expansion slot on a hosting board.

```
( <oid-at150> NAME 'cimSupportsHotPlug'
  DESC 'Boolean indicating whether the Slot supports hot-plug of
      adapter Cards.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at151> NAME 'cimHeightAllowed'
  DESC 'Maximum height of an adapter Card that can be inserted into
      the Slot, in inches.'
```

```
  SUP cim23Float32 SINGLE-VALUE
)

( <oid-at152> NAME 'cimLengthAllowed'
  DESC 'Maximum length of an adapter Card that can be inserted into
      the Slot, in inches.'
  SUP cim23Float32 SINGLE-VALUE
)

( <oid-at153> NAME 'cimMaxDataWidth'
  DESC 'Maximum bus width of adapter Cards that can be inserted
      into this Slot, in bits. If the value is "unknown", enter
      0. If the value is other than 8, 16, 32, 64 or 128, enter
      1. Values are 0, 1, 8, 16, 32, 64, 128.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at154> NAME 'cimVccMixedVoltageSupport'
  DESC 'An array of enumerated integers indicating the Vcc voltage
      supported by this Slot. Values are 0="Unknown", 1="Other",
      2="3.3V", 3="5V".'
  SYNTAX integer
)

( <oid-at155> NAME 'cimVppMixedVoltageSupport'
  DESC 'An array of enumerated integers indicating the Vpp voltage
      supported by this Slot. Values are 0="Unknown", 1="Other",
      2="3.3V", 3="5V", 4="12V".'
  SYNTAX integer
)

( <oid-at156> NAME 'cimThermalRating'
  DESC 'Maximum thermal dissipation of the Slot in milliwatts.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at157> NAME 'cimSpecialPurpose'
  DESC 'Boolean indicating that this Slot is physically unique and
      may hold special types of hardware, e.g. a graphics
      processor slot. If set to TRUE, then the property,
      SpecialPurposeDescription (a string), should specify the
      nature of the uniqueness or purpose of the Slot.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-at158> NAME 'cimPurposeDescription'
  DESC 'A free-form string describing that this Slot is physically
      unique and may hold special types of hardware. This
      property only has meaning when the corresponding boolean
      property, SpecialPurpose, is set to TRUE.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( <oid-at159> NAME 'cimNumber'
  DESC 'The Number property indicates the physical slot number,
      which can be used as an index into a system slot table,
      whether or not that slot is physically occupied.'
  SYNTAX integer SINGLE-VALUE
)

( <oid-at160> NAME 'cimPowered'
  DESC 'A boolean indicating whether the Slot is currently powered
       (TRUE) or not (FALSE).'
  SYNTAX boolean SINGLE-VALUE
```

```
)

( <oid-at161> NAME 'cimOpenSwitch'
  DESC 'A boolean indicating whether the switch state of the Slot is
      currently open (TRUE) or closed (FALSE). This switch state
      determines whether the contents of the Slot can be hot-
      plugged.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-oc82> NAME 'cim23Slot'
  DESC 'The Slot class represents Connectors into which Packages
      are inserted. For example, a PhysicalPackage that is a
      DiskDrive may be inserted into an SCA "Slot". As another
      example, a Card (subclass of PhysicalPackage) may be
      inserted into a 16-, 32-, or 64-bit expansion "Slot" on a
      HostingBoard. PCI or PCMCIA Type III Slots are examples of
      the latter.'
  SUP cim23PhysicalConnector
  MAY ( cimSupportsHotPlug $ cimHeightAllowed $ cimLengthAllowed $
     cimMaxDataWidth $ cimThermalRating $
     cimVccMixedVoltageSupport $ cimVppMixedVoltageSupport $
     cimSpecialPurpose $ cimPurposeDescription $ cimNumber $
     cimPowered $ cimOpenSwitch )
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23Slot.

```
( <oid-oc82> NAME 'cim23SlotContentRule'
  DESC 'The auxiliary classes that may be attached to cim23Slot'
  AUX ( cim23SlotInSlotAuxClass $ cim23AdjacentSlotsAuxClass $
     cim23PackageInSlotAuxClass $ cim23CardInSlotAuxClass )
)
```

3.35 cim23SlotInSlotAuxClass

This class represents the ability of an adapter to extend a slot structure, which enables the slot to support cards that would otherwise be incompatible by interfacing to the slot provided by the adapter. This has many practical uses.

```
( <oid-oc83> NAME 'cim23SlotInSlotAuxClass'
  DESC 'Slots are special types of Connectors into which adapter
      Cards are typically inserted. The SlotInSlot relationship
      represents the ability of a special adapter to extend the
      existing Slot structure to enable otherwise incompatible
      Cards to be plugged into a Frame or HostingBoard. The
      adapter effectively creates a new Slot and can be thought
      of (conceptually) as a Slot in a Slot. This enables Cards
      that would otherwise be physically and/or electrically
      incompatible with the existing Slots to be supported, by
      interfacing to the Slot provided by the adapter. This has
      many practical uses. For example, networking boards are
      very expensive. As new hardware becomes available, Chassis
      and even Card configurations change. To protect the
      investment of their customers, networking vendors will
      manufacture special adapters that enable old Cards to fit
      into new Chassis or HostingBoards and/or new Cards to fit
      into old. This is done using a special adapter that fits
      over one or more existing Slots and presents a new Slot
      into which the Card can plug. Both attributes point to
      cim23Slot objects.'
```

```
  SUP cim23ConnectedToAuxClass AUXILIARY
)
```

3.36 cim23AdjacentSlotsAuxClass

This class describes the layout of slots on a hosting board or adapter card and includes the distance between the slots and whether they are 'shared'.

```
( <oid-at162> NAME 'cimSlotARef'
  DESC 'One of the adjacent Slots.'
  SYNTAX DN
)

( <oid-at163> NAME 'cimSlotBRef'
  DESC 'The "other" adjacent Slot.'
  SYNTAX DN
)

( <oid-at164> NAME 'cimDistanceBetweenSlots'
  DESC 'The distance, in inches, between adjacent Slots.'
  SUP cim23Float32 SINGLE-VALUE
)

( <oid-at165> NAME 'cimSharedSlots'
  DESC 'Slots can be located in close proximity on Hosting Boards
      or other Cards, such that if one of these Slots is
      populated by an adapter Card, the other Slot must be left
      empty. This relationship is indicated by the SharedSlots
      boolean set to TRUE.'
  SYNTAX boolean SINGLE-VALUE
)

( <oid-oc84> NAME 'cim23AdjacentSlotsAuxClass'
  DESC 'AdjacentSlots describes the layout of Slots on a
      HostingBoard or adapter Card. Information like the distance
      between the Slots and whether they are "shared" (if one is
      populated, then the other Slot can not be used), is
      conveyed as properties of the association. Both reference
      attributes point to cim23Slot objects.'
  SUP top AUXILIARY
  MAY ( cimSlotARef $ cimSlotBRef $ cimDistanceBetweenSlots $
     cimSharedSlots )
)
```

3.37 cim23PackageInConnectorAuxClass

This class represents the relationship between cards that are into system connectors for power and/or to transfer data. For example, it would be used to describe the insertion of a daughter card onto another card.

```
( <oid-oc85> NAME 'cim23PackageInConnectorAuxClass'
  DESC 'Adapter cards and other "packaging" are plugged into System
      Connectors for power and/or to transfer data. This
      relationship is defined by PackageInConnector. For example,
      it would be used to describe the insertion of a
      daughtercard onto another Card. Various subclasses of
      PackageInConnector are also defined. PackageInSlot and its
      subclass, CardInSlot, are two examples of
      subclasses. Attribute cimAntecedentRef points to
      cim23PhysicalConnector and attribute cimDependentRef points
      to cim23PhysicalPackage.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.38 cim23PackageInSlotAuxClass

Complex networking devices often are based on chassis, which allow for enhancement and/or augmentation of their base functionality adding new chassis devices, similar to adding cards. This auxiliary class models this capability.

```
( <oid-oc86> NAME 'cim23PackageInSlotAuxClass'
  DESC 'Complex networking devices often are Chassis-based. These
      Chassis allow for enhancement and/or augmentation of their
      base functionality by accepting additional Chassis devices,
      similar to accepting functionality in the form of adding
      Cards. This association models this capability.. Attribute
      cimAntecedentRef points to cim23Slot and attribute
      cimDependentRef points to cim23PhysicalPackage.'
  SUP cim23PackageInConnectorAuxClass AUXILIARY
)
```

3.39 cim23CardInSlotAuxClass

Slots are special types of connectors into which cards are inserted. This relationship of a Card in a Slot is made explicit using this class.

```
( <oid-oc87> NAME 'cim23CardInSlotAuxClass'
  DESC 'Slots are special types of Connectors into which adapter
      Cards are inserted. This relationship of a Card in a Slot
      is made explicit using the CardInSlot
      association. Attribute cimAntecedentRef points to cim23Slot
      and attribute cimDependentRef points to cim23Card.'
  SUP cim23PackageInSlotAuxClass AUXILIARY
)
```

3.40 cim23LinkHasConnectorAuxClass

Cables and links use physical connectors to connect physical elements, which this class explicitly defines.

```
( <oid-oc88> NAME 'cim23LinkHasConnectorAuxClass'
  DESC 'Cables and Links utilize PhysicalConnectors to actually
      "connect" PhysicalElements. This association explicitly
      defines this relationship of Connectors for
      PhysicalLinks. Attribute cimGroupComponentRef points to
      cim23PhysicalLink and attribute cimPartComponentRef points to
      cim23PhysicalConnector.'
  SUP cim22ComponentAuxClass AUXILIARY
)
```

3.41 cim23ConnectorOnPackageAuxClass

Physical packages contain connectors and other physical elements, which this class makes explicit.

```
( <oid-oc89> NAME 'cim23ConnectorOnPackageAuxClass'
  DESC 'PhysicalPackages contain Connectors as well as other
      PhysicalElements. The ConnectorOnPackage association makes
      explicit the containment relationship between Connectors
      and Packages. Attribute cimGroupComponentRef points to
      cim23PhysicalPackage and attribute cimPartComponentRef points
      to cim23PhysicalConnector.'
  SUP cim23ContainerAuxClass AUXILIARY
  MAY ( cimLocationWithinContainer )
)
```

4. References

Request For Comments (RFC) and Internet Draft documents are available from numerous mirror sites.

[1] M. Wahl, T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3)," RFC 2251, Decemeber 1997.

[2] M. Wahl, A. Coulbeck, T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions," RFC 2252, December 1997.

[3] DMTF, "CIM Physcial Model, v2.3," http://www.dmtf.org/spec/cims.html.

[4] DMTF, "LDAP Schema for the DMTF Core CIM v2.3 Model", http://www.dmtf.org/spec/denh.html.

[5] DMTF, "LDAP Mapping Guidelines",

A. Structure Rule definitions

To aid the reader in mapping what structure rules have been defined and referenced, the following table lists sections and documents where they have been defined.

Rule	Structural Class	RDN Attribute	Superior Rules	Defined
<sr10>	cimServicePhilosophyInstance	arrayIndex	<sr24>	2.2.1
<sr11>	cimChassisTypeInstance	arrayIndex	<sr26>	2.2.2
<sr12>	cimMediaTypesInstance	arrayIndex	<sr28>	2.2.3
<sr13>	cimPhysicalLabelsInstance	arrayIndex	<sr32>	2.2.4
<sr14>	cim23Location	orderedCimKeys	*	3.1
<sr16>	cim23MemoryCapacity	orderedCimKeys	*	3.5
<sr\18>	cim23ConfigurationCapacity	orderedCimKeys	*	3.6
<sr20>	cim23ReplacementSet	orderedCimKeys	*	3.7
<sr22>	cim23PhysicalPackage	orderedCimKeys	*	3.9
<sr24>	cim23PhysicalFrame	orderedCimKeys	*	3.11
<sr26>	cim23Chassis	orderedCimKeys	*	3.13
<sr28>	cim23StorageMediaLocation	orderedCimKeys	*	3.20
<sr30>	cim23PhysicalComponent	orderedCimKeys	*	3.21
<sr32>	cim23PhysicalMedia	orderedCimKeys	*	3.26
<sr34>	cim23PhysicalLink	orderedCimKeys	*	3.30
<sr36>	cim23PhysicalConnector	orderedCimKeys	*	3.32

* This rule is corresponds to the mapping of a top level object in CIM. This mapping document does not provide suggestions regarding DIT placement of mapped top-level CIM objects.

B. OID Assignments

The following three tables provides the summary of OID assignments made in this document

B.1 Object Classes

OID	Object Class	Section
	cimServicePhilosophyInstance	2.2.1
	cimChassisTypeInstance	2.2.2
	cimMediaTypesSupportedInstance	2.2.3
	cimPhysicalLabelsInstance	2.2.4
	cim23Location	3.1
	cim23PhysicalElementLocationAuxClass	3.2
	cim23PhysicalCapacity	3.3
	cim23ElementCapacityAuxClass	3.4
	cim23MemoryCapacity	3.5
	cim23ConfigurationCapacity	3.6
	cim23ReplacementSet	3.7
	cim23ParticipatesInSetAuxClass	3.8
	cim23PhysicalPackage	3.9
	cim23ContainerAuxClass	3.10
	cim23PhysicalFrame	3.11
	cim23Rack	3.12
	cim23Chassis	3.13
	cim23ChassisInRackAuxClass	3.14
	cim23PackageInChassisAuxClass	3.15
	cim23DockedAuxClass	3.16
	cim23Card	3.17
	cim23SystemBusCard	3.18
	cim23CardOnCardAuxClass	3.19
	cim23StorageMediaLocation	3.20
	cim23PhysicalComponent	3.21
	cim23PackagedComponentAuxClass	3.22
	cim23Chip	3.23
	cim23PhysicalMemory	3.24
	cim23MemoryOnCardAuxClass	3.25

	cim23PhysicalMedia	3.26
	cim23MemoryWithMediaAuxClass	3.27
	cim23PhysicalMediaInLocationAuxClass	3.28
	cim23PhysicalTape	3.29
	cim23PhysicalLink	3.30
	cim23ElementsLinkedAuxClass	3.31
	cim23PhysicalConnector	3.32
	cim23ConnectedToAuxClass	3.33
	cim23Slot	3.34
	cim23SlotInSlotAuxClass	3.35
	cim23AdjacentSlotsAuxClass	3.36
	cim23PackageInConnectorAuxClass	3.37
	cim23PackageInSlotAuxClass	3.38
	cim23CardInSlotAuxClass	3.39
	cim23LinkHasConnectorAuxClass	3.40
	cim23ConnectorOnPackageAuxClass	3.41

B.2 Attributes

OID	Attribute	Section
	cimServicePhilosophy	2.2.1
	cimServiceDescriptions	2.2.1
	cimChassisTypes	2.2.2
	cimTypeDescriptions	2.2.2
	cimMediaTypesSupported	2.2.3
	cimMediaSizesSupported	2.2.3
	cimPhysicalLabels	2.2.4
	cimLabelStates	2.2.4
	cimLabelFormats	2.2.4
	cimPhysicalPosition	3.1
	cimAddress	3.1
	cimPhysicalLocationRef	3.2
	cimCapacityRef	3.4
	cimMemoryType	3.5
	cimMinimumMemoryCapacity	3.5
	cimMaximumMemoryCapacity	3.5
	cimObjectType	3.6
	cimOtherTypeDescription	3.6
	cimMimimumCapacity	3.6

	cimMaximumCapacity	3.6
	cimIncrement	3.6
	cimSetRef	3.8
	cimRemovable	3.9
	cimReplaceable	3.9
	cimHotSwappable	3.9
	cimHeight	3.9
	cimDepth	3.9
	cimWidth	3.9
	cimWeight	3.9
	cimLocationWithinContainer	3.10
	cimCableManagementStrategy	3.11
	cimLockPresent	3.11
	cimAudibleAlarm	3.11
	cimVisibleAlarm	3.11
	cimSecurityBreach	3.11
	cimBreachDescription	3.11
	cimIsLocked	3.11
	cimTypeOfRack	3.12
	cimNumberOfPowerCords	3.13
	cimCurrentRequiredOrProduced	3.13
	cimHeatGeneration	3.13
	cimBottomU	3.14
	cimHostingBoard	3.17
	cimSlotLayout	3.17
	cimRequiresDaughterBoard	3.17
	cimSpecialRequirements	3.17
	cimRequirementsDescription	3.17
	cimOperatingVoltages	3.17
	cimBusType	3.18
	cimBusWidth	3.18
	cimMountOrSlotDescription	3.19
	cimLocationType	3.20
	cimLocationCoordinates	3.20
	cimMediaCapacity	3.20
	cimFormFactor	3.23
	cimTotalWidth	3.24
	cimDataWidth	3.24

	cimSpeed	3.24
	cimCapacity	3.24
	cimBankLabel	3.24
	cimPositionInRow	3.24
	cimInterleavePosition	3.24
	cimMediaType	3.26
	cimMediaDescription	3.26
	cimWriteProtectOn	3.26
	cimCleanerMedia	3.26
	cimMediaSize	3.26
	cimMaxMounts	3.26
	cimDualSided	3.26
	cimTapeLength	3.29
	cimUnloadAnywhere	3.29
	cimMaxLength	3.30
	cimLength	3.30
	cimWired	3.30
	cimConnectorPinout	3.32
	cimConnectorType	3.32
	cimSupportsHotPlug	3.34
	cimHeightAllowed	3.34
	cimLengthAllowed	3.34
	cimMaxDataWidth	3.34
	cimVccMixedVoltageSupport	3.34
	cimVppMixedVoltageSupport	3.34
	cimThermalRating	3.34
	cimSpecialPurpose	3.34
	cimPurposeDescription	3.34
	cimNumber	3.34
	cimPowered	3.34
	cimOpenSwitch	3.34
	cimSlotARef	3.36
	cimSlotBRef	3.36
	cimDistanceBetweenSlots	3.36
	cimSharedSlots	3.36

B.3 Nameforms

OID	Nameform	Section

	cimServicePhilosophyInstanceNameForm	2.2.1
	cimChassisTypeInstanceNameForm	2.2.2
	cimMediaTypesSupportedInstanceNameForm	2.2.3
	cimPhysicalLabelsInstanceNameForm	2.2.4
	cim23LocationNameForm	3.1
	cim23MemoryCapacityNameForm	3.5
	cim23ConfigurationCapacityNameForm	3.6
	cim23ReplacementSetNameForm	3.7
	cim23PhysicalPackageNameForm	3.9
	cim23PhysicalFrameNameForm	3.11
	cim23ChassisNameForm	3.13
	cim23StorageMediaLocationNameForm	3.20
	cim23PhysicalComponentNameForm	3.21
	cim23PhysicalMediaNameForm	3.26
	cim23PhysicalLinkNameForm	3.30
	cim23PhysicalConnectorNameForm	3.32