Specification *DSP0101*

DMTF Core CIM v2.3 LDAP Mapping

April 24, 2000

Abstract

This document presents an LDAP core schema for the DMTF CIM Core model version 2.3 [3].

Table of Contents

1. Introduction

This document presents an LDAPv3 [1,2] schema for the DMTF CIM Core 2.3 Model [3]. Associations are mapped using a combination of auxiliary classes and DIT structure rules. The content and structure rules provided here are suggestions and may be modified as needed to support a particular directory structure. Directory administrators do not need to subclass/instantiate all of the classes in this schema. They are free to choose the subset that meets their particular needs. In the mapping of properties to attributes, syntax object identifiers have been replaced by textual names. The correspondence between names and OIDs is:

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
generalizedTime	1.3.6.1.4.1.1466.115.121.1.24
integer	1.3.6.1.4.1.1466.115.121.1.27

2. LDAP Mapping Considerations

2.1 Differences from the Core CIM Model

This mapping differs from the core CIM model in that the hierarchy of classes beginning with CIM_StatisticalInformation or CIM_Statistics have not been mapped, nor has the CIM_DependencyContext association been mapped.

This mapping uses DIT containment to map weak CIM associations. LDAP allows structural rules only for structural classes. The CIM associations HostedService, HostedAccessPoint, and SystemDevice are weak associations between abstract classes so while these associations are not mapped, the DIT containment rules will be instantiated for structural subclasses of these abstract classes.

CIM Association	DIT Superior Class	DIT Subordinate Class
HostedService	cim23System	cim23Service
HostedAccessPoint	cim23System	cim23ServiceAccessPoint
SystemDevice	cim23System	cim23LogicalDevice

2.2 Inheritance Assumptions

2.2.1 General

This schema mapping is based on the assumption that name form rules structural rules, and content rules are inherited by both auxiliary and structural sub-classes. For implementations where this is not the case, extra rules will need to be specified.

On the other hand, the attributes contained in auxiliary classes in this mapping are based on the assumption that inheritance does not occur between auxiliary classes (even though such inheritance is shown in the auxiliary class definitions). For implementations that do support this type of inheritance, the schema definitions will be simpler.

2.2.2 Object Classes

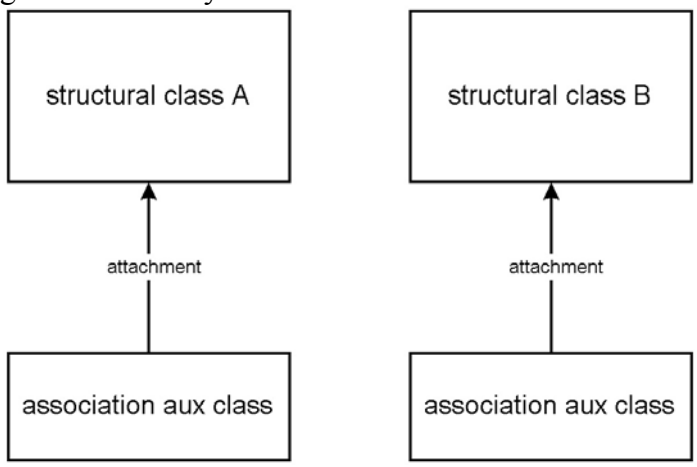
For simplicity, updated object classes will inherit only a single parent. While the LDAP standards do not preclude specifying multiple inheritance for object classes, it is doubtful that such a specification would find use in general implementations.

2.3 Helper Classes

2.3.1 cimAssociationInstance

This object class is used as an alternative attachment point for association auxiliary class information. Each association instance of an object is represented by an instance of cimAssociationInstance immediately subordinate to the object's entry. The association auxiliary class is then attached to the cimAssociationInstance entry. Therefore, cimAssociationInstance may have an CIM association auxiliary class attached to it. This alternative is required, for example, when an object engages in multiple instances of a particular type of association, in which case it would not be possible to attach multiple auxiliary classes of the same type to the object's entry. For simplicity, symmetric mapping is assumed. This leads to two possibilities: the auxiliary classes for an instance of a particular CIM association class are attached to the structural class on both sides of the association as shown in Figure 1, or they are attached to cimAssociationInstance objects on both sides of the association as shown in Figure 2.

Figure 1. Auxiliary Class Attachment to Structural Classes



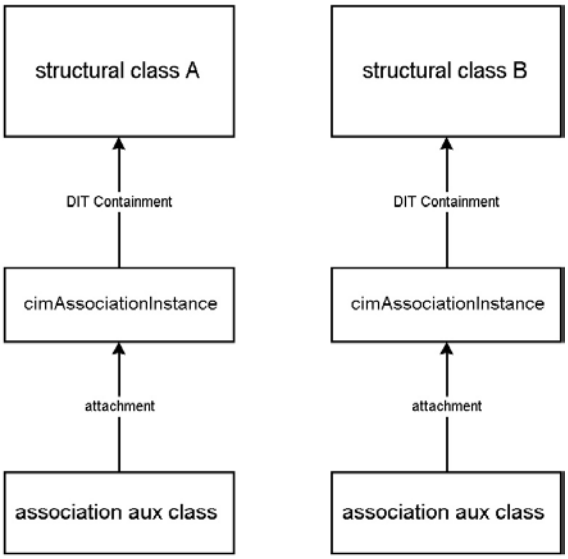
Because these classes are subordinates, they use cimAssociationName as their RDN. So that directory implementers may use their own unique names in cimAssociationName, cimAssociationTypeName is also defined that may store extra information. This information is intended to hold the CIM Association Type (e.g. "CIM_VLANFor") to aid when searching.

```
( 1.3.6.1.4.1.412.100.1.2.3 NAME 'cimAssociationName'
  DESC 'The RDN of this association instance'
  EQUALITY caseIgnoreMatch SUBSTR caseIgnoreSubstringsMatch
```

```
  SYNTAX DirectoryString SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.1.2.4 NAME 'cimAssociationTypeName'
  DESC 'support storing extra information about the
        association type'
  SYNTAX DirectoryString SINGLE-VALUE
)
```

Figure 2. Auxiliary Class Attachment to cimAssociationInstance




```
( 1.3.6.1.4.1.412.100.1.1.1.1 NAME 'cimAssociationInstance'
  SUP top
  MUST ( cimAssociationName )
  MAY ( cimAssociationTypeName )
)

( 1.3.6.1.4.1.412.100.1.3.1.1 NAME 'cimAssociationInstanceNameForm'
  OC cimAssociationInstance
  MUST ( cimAssociationName )
)

( <sr5> NAME 'cimAssociationInstanceStructureRule'
  FORM cimAssociationInstanceNameForm
  SUP ( <sr1> <sr2> <sr3> <sr4> )
)
```

2.3.2 cimOtherIdentifyingInfoInstance

CIM defines the concept of an ordered array, which LDAP does not support. In the core CIM model, indexed arrays are only used in two abstract classes (CIM_ComputerSystem and CIM_LogicalDevice) to tie the values of two property arrays together. In the LDAP mapping, these properties are replaced with separate instances of cimOtherIdentifyingInfoInstance that each contain a single pair of attribute values and are DIT contained by the parent class. The attribute cimOtherIdentifyingInfo is already defined below and reused here and the attribute arrayIndex is defined as the RDN for this

class. Finally, the structure rule is provided as a template to be filled in with structure rule pointers to structural rules defined for sub-classes of cim23ComputerSystem and cim23LogicalDevice.

```
( 1.3.6.1.4.1.412.100.1.2.5 NAME 'arrayIndex'
  DESC 'the index of this child'
  SYNTAX DirectoryString
  EQUALITY caseIgnoreMatch
)

( 1.3.6.1.4.1.412.100.2.2.23 NAME 'cimIdentifyingDescription'
  DESC 'A free-form string providing explanation and
        details behind the entries in the OtherIdentifyingInfo
        attribute.'
  SYNTAX DirectoryString
)

( 1.3.6.1.4.1.412.100.2.1.1.43
  NAME 'cimOtherIdentifyingInfoInstance'
  DESC 'helper class to tie indexed arrays in core model together'
  SUP top
  MUST ( arrayIndex )
  MAY ( cimOtherIdentifyingInfo $ cimIdentifyingDescription )
)

( 1.3.6.1.4.1.412.100.2.3.1.5
  NAME 'cimOtherIdentifyingInfoInstanceNameForm'
  OC cimOtherIdentifyingInfoInstance
  MUST ( arrayIndex )
)

( <sr6> NAME 'cimOtherIdentifyingInfoInstanceStructureRule'
  FORM cimOtherIdentifyingInfoInstanceNameForm
)
```

2.4 Naming considerations

To support naming in the LDAP mapping of the core schema, the attributes orderedCimKeys and orderedCimModelPath are defined. They provide the RDN for directory implementations.

```
( 1.3.6.1.4.1.412.100.1.2.1 NAME 'orderedCimKeys'
  DESC 'The model path for the instance (without propagated
        keys). May be used as an RDN'
  SYNTAX DirectoryString SINGLE-VALUE
  EQUALITY octetStringMatch
)

( 1.3.6.1.4.1.412.100.1.2.2 NAME 'orderedCimModelPath'
  DESC 'The model path for the instance (with propagated keys). May
        be used as an RDN'
  SYNTAX DirectoryString SINGLE-VALUE
  EQUALITY octetStringMatch
)
```

The value of these attributes are constructed by ordering the CIM keys [formatted as "**<className>**.**<key>**=**<value>**[,**<key>**=**<value>**]*"] of the object in the US-ASCII

collation order of the property types. If the DIT structure follows the CIM namespace structure, orderedCimKeys is used and does not include propagated keys. If the DIT structure is different that the CIM namespace structure, then orderedCimModelPath is used as the naming attribute and includes all keys.

2.5 Syntax Conversions

This section discusses specific conversions needed for the core schema. Other mappings may define additional conversion procedures.

2.5.1 CIM String and LDAP DirectoryString

Strings in CIM are stored as UCS-2 characters, while LDAP DirectoryStrings are stored in UTF-8 format. Use RFC 2279 [4] to convert between these two formats.

2.5.2 CIM DateTime and LDAP GeneralizedTime

CIM DateTime is used to store both timestamps and intervals in UCS-2. LDAP GeneralizedTime stores timestamps in a subset of UTF-8. In addition, timezone offset in DateTime are specified in minutes, while GeneralizedTime uses hours and minutes. Because the core schema only stores timestamps, GeneralizedTime may be used. Time interval conversions will be handled when necessary. Finally, DateTime allows insignificant parts of the date string to be replaced by asterisks ('*'). To convert from DateTime to GeneralizedTime, the following steps are used:

1. Use RFC 2279 to convert from UCS-2 to UTF-8.
2. Any asterisks in DateTime starting from the rightmost position (microseconds) up to the first date field may be omitted. Any remaining asterisk filled fields must be replaced by zeros.
3. If the UTC offset is +000 it is replaced with a "Z". Otherwise the UTC offset is translated from minutes to hours and minutes format.

To convert from GeneralizedTime to DateTime, the following steps are used:

1. Zero pad or truncate the decimal portion of the seconds to be exactly 6 digits. If there are no decimal seconds specified then the decimal point "." and six asterisks must be used.
2. If the value is in UTC, that is it is followed by a "Z" the "Z" is replaced with +000. Otherwise, the UTC offset is translated from hhmm format to mmm format.
3. If a comma is used as the decimal separator - replace it with a period.
4. Perform character set translation as required.

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. While this approach was chosen to minimize the number of DN pointers stored in the schema, some pointer dereferencing is necessary.

3.1 cim23ManagedElement

This abstract class provides a base for non-association classes in CIM. Its addition is one of the major changes between CIM v2.2 and CIM v2.3.

```
( 1.3.6.1.4.1.412.100.2.2.1 NAME 'cimCaption'
  DESC 'The Caption property is a short textual description
     (one-line string) of the object.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.2 NAME 'cimDescription'
  DESC 'The Description property provides a textual description of
      the object.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.44 NAME 'cim23ManagedElement'
  DESC 'ManagedElement is an abstract class that provides a common
       superclass (or top of the inheritance tree) for the
       non-association classes in the CIM Schema.'
  SUP top ABSTRACT
  MAY ( cimCaption $ cimDescription $ orderedCimKeys $
       orderedCimModelPath )
)
```

3.2 cim23ManagedSystemElement

This is the base class for the system element hierarchy. Any distinguishable component of a system is a candidate for inclusion in this class. Examples of this are software components, such as files and devices, such as disk drives and controllers, and physical components such as chips and cards.

```
( 1.3.6.1.4.1.412.100.2.2.3 NAME 'cimInstallDate'
  DESC 'A datetime value indicating when the object was
      installed.'
  SYNTAX generalizedTime SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.4 NAME 'cimName'
  DESC 'The Name property defines the label by which the object is
      known. When subclassed, the Name property can be overridden
      to be a key property.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.5 NAME 'cimStatus'
  DESC 'A string indicating the current status of the
      object. Various operational and non-operational statuses
      are defined. Operational statuses are "OK", "Degraded",
      "Stressed" and "Pred Fail". "Stressed" shows that the
      Element is functioning, but needs attention. Examples of
      "Stressed" states are overload, overheated, etc. The
      condition "Pred Fail" (failure predicted) shows that
      an Element is functioning properly but predicting a
      failure soon. An example is a SMART-enabled hard
      drive. Non-operational statuses can also be specified.
      These are "Error", "NonRecover", "Starting",
        "Stopping" and "Service". "NonRecover" shows that a
```

```
        non-recoverable error has occurred. "Service" describes an
        Element being configured, maintained or cleaned, or
        otherwise administered. This status could apply during
        mirror-resilvering of a disk, reload of a user permissions
        list, or other administrative task. Not all such work is
        on-line, yet the Element is neither "OK" nor in
        another state.   Values are "OK", "Error", "Degraded",
        "Unknown", "Pred Fail", "Starting", "Stopping", "Service",
        "Stressed", "NonRecover".'
    SYNTAX DirectoryString{10} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.1 NAME 'cim23ManagedSystemElement'
    DESC 'CIM_ManagedSystemElement is the base class for the System
          Element hierarchy. Membership Criteria: Any distinguishable
          component of a System is a candidate for inclusion in this
          class. Examples: software components, such as files; and
          devices, such as disk drives and controllers, and physical
          components such as chips and cards.'
    SUP cim23ManagedElement ABSTRACT
    MAY ( cimInstallDate $ cimName $ cimStatus )
)
```

3.3 cim23Collection

This abstract class provides a common superclass for classes that represent collections of managed elements.

```
( 1.3.6.1.4.1.412.100.2.1.2.45 NAME 'cim23Collection'
    DESC 'Collection is an abstract class that provides a
          common superclass for data elements that represent
          collections of ManagedElements and its subclasses.'
    SUP cim23ManagedElement ABSTRACT
)
```

3.4 cim23CollectionOfMSEs

This object allows the grouping of cim23ManagedSystemElement objects for associating settings and configurations. It is abstract to require further definition and semantic refinement in subclasses. As this object does not carry any state or status information, it only represents a grouping or 'bag' of elements. So, it is incorrect to subclass groups that have state/status from this class - an example is cim23RedundancyGroup (which is correctly subclassed from cim23LogicalElement).
Collections typically aggregate 'like' objects, and represent an optimization. Without collections, one is forced to define individual associations, to tie settings and configuration objects to individual cim23ManagedSystemElements. There may be much duplication in assigning the same setting to multiple objects. In addition, using this object allows the determination that the setting and configuration associations are indeed the same for the collection's members. This information would otherwise be obtained by defining the collection in a proprietary way, and then querying the associations to determine if the collection set is completely covered.

```
( 1.3.6.1.4.1.412.100.2.2.6 NAME 'cimCollectionID'
```

```
  DESC 'The identification of the Collection object. When
        subclassed, the CollectionID property can be overridden
        to be a Key property.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.2 NAME 'cim23CollectionOfMSEs'
  DESC 'The CollectionOfMSEs object allows the grouping of
        ManagedSystemElements for the purposes of associating
        Settings and Configurations. It is abstract to require
        further definition and semantic refinement in
        subclasses. The CollectionOfMSEs object does not carry any
        state or status information, but only represents a grouping
        or "bag" of Elements. Therefore, it is incorrect to
        subclass groups that have state/status from
        CollectionOfMSEs - an example is cim23RedundancyGroup (which
        is correctly subclassed from LogicalElement). Collections
        typically aggregate "like" objects, and represent an
        optimization. Without Collections, one is forced to define
        individual ElementSetting and ElementConfiguration
        associations, to tie Settings and Configuration objects to
        individual ManagedSystemElements. There may be much
        duplication in assigning the same Setting to multiple
        objects. In addition, using the Collection object allows
        the determination that the Setting and Configuration
        associations are indeed the same for the Collection\'s
        members. This information would otherwise be obtained by
        defining the Collection in a proprietary way, and then
        querying the ElementSetting and ElementConfiguration
        associations to determine if the Collection set is
        completely covered.'
  SUP cim23Collection ABSTRACT
  MAY ( cimCollectionID )
)
```

3.5 cim23MemberOfCollectionAuxClass

This auxiliary class establishes membership of cim23ManagedElement objects in a collection.

```
( 1.3.6.1.4.1.412.100.2.2.7 NAME 'cimCollectionRef'
  DESC 'The grouping or "bag" object that represents the
        Collection.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.8 NAME 'cimMemberRef'
  DESC 'The members of the Collection.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.2.46
  NAME 'cim23MemberOfCollectionAuxClass'
  DESC 'CIM_MemberOfCollection is an aggregation used to establish
        membership of ManagedElements in a
        Collection. cimCollectionRef points to a cim23Collection
```

```
        object and cimMemberRef points to a cim23ManagedElement
        object.'
  SUP top AUXILIARY
  MAY ( cimCollectionRef $ cimMemberRef )
)
```

3.6 cim23CollectedMSEsAuxClass

This auxiliary class represents a generic association used to establish the members of the grouping object, cim23CollectionOfMSEs.

```
( 1.3.6.1.4.1.412.100.2.1.2.3 NAME 'cim23CollectedMSEsAuxClass'
  DESC 'cimCollectedMSEs is a generic association used to
        establish the members of the grouping object,
        CollectionOfMSEs.  Attribute cimCollectionRef points to
        cim23CollectionOfMSEs and cimMemberRef points to
        cim23ManagedSystemElement.'
  SUP cim23MemberOfCollectionAuxClass AUXILIARY
)
```

3.7 cim22CollectedCollectionsAuxClass

This association represents that a CollectionOfMSEs may itself be contained in another CollectionOfMSEs object.

```
( 1.3.6.1.4.1.412.100.2.2.9 NAME 'cimCollectionInCollectionRef'
  DESC 'The "collected" Collection.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.4
  NAME 'cim22CollectedCollectionsAuxClass'
  DESC 'cim22CollectedCollections is an aggregation association
        representing that a CollectionOfMSEs may itself be
        contained in a CollectionOfMSEs. Both attributes point to
        cim22CollectionOfMSEs or cim23CollectionOfMSEs.'
  SUP top AUXILIARY
  MAY ( cimCollectionRef $ cimCollectionInCollectionRef )
)
```

3.8 cim23PhysicalElement

This class acts as the base class for any component of a system that has a distinct physical identity. Instances of this class can be defined in terms of labels that can be physically attached to the object. All processes, files, and logical devices are considered not to be physical elements. For example, it is not possible to attach a label to a modem. It is only possible to attach a label to the card that implements the modem. The same card could also implement a LAN adapter. This is an example of a single physical element (the card) hosting more than one logical device.

```
( 1.3.6.1.4.1.412.100.2.2.10 NAME 'cimCreationClassName'
  DESC 'CreationClassName shows the name of the class or the
        subclass used in the creation of an instance. When used
        with the other key properties of this class, this property
```

```
        allows all instances of this class and its subclasses to be
        uniquely identified.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.11 NAME 'cimTag'
  DESC 'An arbitrary string that uniquely identifies the
        PhysicalElement and serves as the Element"s key.  The Tag
        property can contain information such as asset tag or
        serial number data.  The key for PhysicalElement is placed
        high in the object hierarchy to independently
        identify the hardware/entity, regardless of physical
        placement in or on Cabinets, Adapters, etc. For example, a
        hotswappable or removeable component may be taken from its
        containing (scoping) Package and be temporarily unused. The
        object still exists - and may even be inserted
        into a different scoping container. Therefore, the key for
        PhysicalElement is an arbitrary string and is defined
        independently of any placement or location-oriented
        hierarchy.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.12 NAME 'cimManufacturer'
  DESC 'The name of the organization responsible for producing the
        PhysicalElement. This may be the entity from whom the
        Element is purchased, but this is not necessarily true. The
        latter information is contained in the Vendor property of
        cim22Product.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.13 NAME 'cimModel'
  DESC 'The name by which the PhysicalElement is generally known.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.14 NAME 'cimSKU'
  DESC 'The stock keeping unit number for this PhysicalElement.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.15 NAME 'cimSerialNumber'
  DESC 'A manufacturer-allocated number used to identify the
        PhysicalElement.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.16 NAME 'cimVersion'
  DESC 'A string indicating the version of the PhysicalElement.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.17 NAME 'cimPartNumber'
  DESC 'The part number assigned by the organization responsible
        for producing or manufacturing the PhysicalElement.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
```

```
)

( 1.3.6.1.4.1.412.100.2.2.18 NAME 'cimOtherIdentifyingInfo'
  DESC 'OtherIdentifyingInfo captures additional data, beyond asset
        tag information, that could be used to identify a
        PhysicalElement. One example is bar code data associated
        with an Element that also has an asset tag. Note that if
        only bar code data is available and is unique/able to be
        used as an Element key, this property would be NULL and the
        bar code data used as the class key, in the Tag property.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.19 NAME 'cimPoweredOn'
  DESC 'Boolean indicating that the PhysicalElement is powered on
        (TRUE), or is currently off (FALSE).'
  SYNTAX boolean SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.69 NAME 'cimManufactureDate'
  DESC 'Date that this PhysicalElement was manufactured.'
  SYNTAX generalizedTime SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.5 NAME 'cim23PhysicalElement'
  DESC 'Subclasses of cim23PhysicalElement define any component of a
        System that has a distinct physical identity. Instances of
        this class can be defined in terms of labels that can be
        physically attached to the object. All Processes, Files,
        and LogicalDevices are considered not to be
        PhysicalElements. For example, it is not possible to attach
        a label to a modem. It is only possible to attach a label
        to the card that implements the modem. The same card could
        also implement a LAN adapter. These are tangible Managed
        System Elements (usually hardware items) that have a
        physical manifestation of some sort. A Managed System
        Element is not necessarily a discrete component. For
        example, it is possible for a single Card (which is a type
        of Physical Element) to host more than one Logical
        Device. The card would be represented by a single Physical
        Element associated with multiple Logical Devices.'
  SUP cim23ManagedSystemElement ABSTRACT
  MAY ( cimCreationClassName $ cimManufacturer $ cimModel $ cimSKU $
       cimSerialNumber $ cimTag $ cimVersion $ cimPartNumber $
       cimOtherIdentifyingInfo $ cimPoweredOn $
       cimManufactureDate )
)
```

3.9 cim23LogicalElement

This class is the base class for all the components of a system that represent abstract system components, such as files, processes, or system capabilities as logical devices.

```
( 1.3.6.1.4.1.412.100.2.1.2.6 NAME 'cim23LogicalElement'
  DESC 'cim23LogicalElement is a base class for all the components
        of a System that represent abstract system components, such
        as Files, Processes, or system capabilities as Logical
```

```
        Devices.'
  SUP cim23ManagedSystemElement ABSTRACT
)
```

3.10 cim22LogicalIdentityAuxClass

This auxiliary class represents an abstract and generic association, showing that two LogicalElements represent different aspects of the same underlying entity. This relationship conveys what could be defined with multiple inheritance. It is restricted to the 'logical' aspects of a ManagedSystemElement. In most scenarios, the identity relationship is determined by the equivalence of keys or some other identifying properties of the related elements. The association should only be used in well understood scenarios. This is why the association is abstract - allowing more concrete definition and clarification in subclasses.

```
( 1.3.6.1.4.1.412.100.2.2.20 NAME 'cimSystemElementRef'
   DESC 'SystemElement represents one aspect of the
         LogicalElement.'
   SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.21 NAME 'cimSameElementRef'
  DESC 'SameElement represents an alternate aspect of the System
        entity.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.7 NAME 'cim22LogicalIdentityAuxClass'
  DESC 'cim22LogicalIdentityAuxClass is an abstract and generic
        association, indicating that two LogicalElements represent
        different aspects of the same underlying entity. This
        relationship conveys what could be defined with multiple
        inheritance. It is restricted to the "logical" aspects of a
        ManagedSystemElement. In most scenarios, the Identity
        relationship is determined by the equivalence of Keys or
        some other identifying properties of the related
        Elements. The association should only be used in well
        understood scenarios. This is why the association is
        abstract - allowing more concrete definition and
        clarification in subclasses. One scenario where
        this relationship is reasonable is to represent that a
        Device is both a "bus" entity and a "functional"
        entity. For example, a Device could be both a USB (bus) and
        a Keyboard (functional) entity. Both attributes point to
        cim22LogicalElement or cim23LogicalElement objects.'
  SUP top AUXILIARY
  MAY ( cimSystemElementRef $ cimSameElementRef )
)
```

3.11 cim23Configuration

This object allows the grouping of sets of parameters (defined in cim23Setting objects) and dependencies for one or more managed system elements. The configuration object represents a certain behavior, or a desired functional state for the managed system elements. The desired functional state is typically driven by external requirements such as

time or location. For example, to connect to a Mail System from 'home', a dependency on a modem exists, but a dependency on a network adapter exists at 'work'. Settings for the pertinent logical devices can be defined and aggregated by the configuration. Therefore, two 'Connect to Mail' configurations may be defined grouping the relevant dependencies and setting objects.

```
( 1.3.6.1.4.1.412.100.2.1.2.8 NAME 'cim23Configuration'
  DESC 'The Configuration object allows the grouping of sets of
        parameters (defined in Setting objects) and dependencies
        for one or more ManagedSystemElements. The Configuration
        object represents a certain behavior, or a desired
        functional state for the ManagedSystemElements. The desired
        functional state is typically driven by external
        requirements such as time or location. For example, to
        connect to a Mail System from "home", a dependency on a
        modem exists, but a dependency on a network adapter exists
        at "work". Settings for the pertinent LogicalDevices (in
        this example, POTSModem and NetworkAdapter) can be defined
        and aggregated by the Configuration. Therefore, two
        "Connect to Mail" Configurations may be defined grouping
        the relevant dependencies and Setting objects.'
  SUP cim23ManagedElement
  MAY ( cimName )
)

( 1.3.6.1.4.1.412.100.2.3.2.1 NAME 'cim23ConfigurationNameForm1'
  OC cim23Configuration
  MUST ( orderedCimKeys )
)

( <sr1> NAME 'cim23ConfigurationStructureRule1'
  FORM cim23ConfigurationNameForm1
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23Configuration.

```
( 1.3.6.1.4.1.412.100.2.1.2.8 NAME 'cim23ConfigurationContentRule'
  DESC 'The auxiliary classes that may be attached to
        cim23Configuration'
  AUX ( cim22ConfigurationComponentAuxClass $
       cim22ElementConfigurationAuxClass $
       cim22CollectionConfigurationAuxClass $
       cim22SettingContextAuxClass $ cim22DependencyAuxClass $
       cim23MemberOfCollectionAuxClass )
)
```

3.12 cim22ConfigurationComponentAuxClass

This association aggregates 'lower-level' configuration objects into a \'high-level' configuration. This enables the assembly of complex configurations by grouping together simpler ones.

```
( 1.3.6.1.4.1.412.100.2.2.24 NAME 'cimConfigGroupRef'
  SYNTAX DN
```

```
)

( 1.3.6.1.4.1.412.100.2.2.25 NAME 'cimConfigComponentRef'
  DESC 'A Configuration that is part of a "higher-level"
        Configuration.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.9
  NAME 'cim22ConfigurationComponentAuxClass'
  DESC 'ConfigurationComponent aggregates "lower-level"
        Configuration objects into a "high-level"
        Configuration. This enables the assembly of complex
        Configurations by grouping together simpler ones. For
        example, a logon policy for the United States could consist
        of two Configuration groups, one for the east coast and one
        for the west coast. Each of these could in turn consist of
        multiple Configurations to handle different aspects of the
        logon process. Both attributes point to cim22Configuration
        or cim23Configuration objects.'
  SUP top AUXILIARY
  MAY ( cimConfigGroupRef $ cimConfigComponentRef )
)
```

3.13 cim22ElementConfigurationAuxClass

This association relates a configuration object to one or more managed system elements. The configuration object represents a certain behavior, or a desired functional state for the associated managed system elements.

```
( 1.3.6.1.4.1.412.100.2.2.26 NAME 'cimElementRef'
  DESC 'The ManagedSystemElement.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.27 NAME 'cimConfigurationRef'
  DESC 'The Configuration object that groups the Settings and
        dependencies associated with the ManagedSystemElement.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.10
  NAME 'cim22ElementConfigurationAuxClass'
  DESC 'This association relates a Configuration object to one or
        more ManagedSystemElements. The Configuration object
        represents a certain behavior, or a desired functional
        state for the associated ManagedSystemElements. Attribute
        cimElementRef points to cim22ManagedSystemElement or
        cim23ManagedSystemElement and attribute cimConfigurationRef
        points to cim22Configuration or cim23Configuration.'
  SUP top AUXILIARY
  MAY ( cimElementRef $ cimConfigurationRef )
)
```

3.14 cim22CollectionConfigurationAuxClass

This auxiliary class relates a Configuration object to one or more CollectionOfMSEs objects. The Configuration object represents a certain behavior, or a desired functional state for the associated collection.

```
( 1.3.6.1.4.1.412.100.2.1.1.11
  NAME 'cim22CollectionConfigurationAuxClass'
  DESC 'This association relates a Configuration object to one or
        more CollectionOfMSEs objects. The Configuration object
        represents a certain behavior, or a desired functional
        state for the associated Collection. Attribute
        cimCollectionRef points to cim22CollectionOfMSEs or
        cim23CollectionOfMSEs and attribute cimConfigurationRef
        points to cim22Configuration or cim23Configuration.'
  SUP top AUXILIARY
  MAY ( cimCollectionRef $ cimConfigurationRef )
)
```

3.15 cim23Setting

This class represents configuration-related and operational parameters for one or more managed system element(s). A managed system element may have multiple setting objects associated with it. The current operational values for an element's parameters are reflected by properties in the element itself or by properties in its associations. These properties do not have to be the same values present in the setting object. For example, a modem may have a setting baud rate of 56Kb/sec but be operating at 19.2Kb/sec.

```
( 1.3.6.1.4.1.412.100.2.2.28 NAME 'cimSettingID'
  DESC 'The identifier by which the Setting object is known.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.12 NAME 'cim23Setting'
  DESC 'The Setting class represents configuration-related and
        operational parameters for one or more
        ManagedSystemElement(s). A ManagedSystemElement may have
        multiple Setting objects associated with it. The current
        operational values for an Element\'s parameters are
        reflected by properties in the Element itself or by
        properties in its associations. These properties do not
        have to be the same values present in the Setting
        object. For example, a modem may have a Setting baud rate
        of 56Kb/sec but be operating at 19.2Kb/sec.'
  SUP cim23ManagedElement ABSTRACT
  MAY ( cimSettingID )
)
```

3.16 cim22ElementSettingAuxClass

This auxiliary class represents the association between managed system elements and the setting class(es) defined for them.

```
( 1.3.6.1.4.1.412.100.2.2.29 NAME 'cimSettingRef'
  DESC 'The Setting object associated with the
        ManagedSystemElement.'
```

```
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.13 NAME 'cim22ElementSettingAuxClass'
  DESC 'ElementSetting represents the association between
        ManagedSystemElements and the Setting class(es) defined for
        them. Attribute cimElementRef points to
        cim22ManagedSystemElement or cim23ManagedSystemElement and
        attribute cimSettingRef points to cim22Setting or
        cim23Setting.'
  SUP top AUXILIARY
  MAY ( cimElementRef $ cimSettingRef )
)
```

3.17 cim22DefaultSettingAuxClass

This auxiliary class represents the association between a ManagedSystemElement and the single Setting class that is defined to be the default setting for this element.

```
( 1.3.6.1.4.1.412.100.2.1.1.14 NAME 'cim22DefaultSettingAuxClass'
  DESC 'DefaultSetting represents the association between a
        ManagedSystemElement and the single Setting class that is
        defined to be the default setting for this
        Element. Attribute cimElementRef points to
        cim22ManagedSystemElement or cim23ManagedSystemElement and
        attribute cimSettingRef points to cim22Setting or
        cim23Setting.'
  SUP cim22ElementSettingAuxClass AUXILIARY
)
```

3.18 cim22SettingContextAuxClass

This auxiliary class associates a setting with one or more configuration objects. For example, a network adapter's settings could change based on the site/network to which its hosting computer system is attached.

```
( 1.3.6.1.4.1.412.100.2.2.30 NAME 'cimContextRef'
  DESC 'The Configuration object that aggregates the Setting.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.15 NAME 'cim22SettingContextAuxClass'
  DESC 'This relationship associates Configuration objects with
        Setting objects. For example, a NetworkAdapter"s Settings
        could change based on the site/network to which its hosting
        ComputerSystem is attached. Here, the
        ComputerSystem would have two different Configuration
        objects, corresponding to the differences in network
        configuration for the two network segments. Configuration A
        would aggregate a Setting object for the NetworkAdapter
        when operating on segment "ANet", whereas Configuration B
        would aggregate a different NetworkAdapter Setting object,
        specific to segment "BNet". Note that many Settings of the
        computer are independent of the network Configuration. For
        example, both Configurations A and B would aggregate the
```

```
        same Setting object for the ComputerSystem\`s
        MonitorResolution.  Attribute cimContextRef points to
        cim22Configuration or cim23Configuration and attribute
        cimSettingRef points to cim22Setting or cim23Setting.'
  SUP top AUXILIARY
  MAY ( cimContextRef $ cimSettingRef )
)
```

3.19 cim22CollectionSettingAuxClass

This auxiliary class represents the association between a CollectionOfMSEs class and the Setting class(es) defined for them.

```
( 1.3.6.1.4.1.412.100.2.1.1.16 NAME 'cim22CollectionSettingAuxClass'
  DESC 'CollectionSetting represents the association between a
        CollectionOfMSEs class and the Setting class(es) defined
        for them. Attribute cimCollectionRef points to
        cim22CollectionOfMSEs or cim23CollectionOfMSEs and attribute
        cimSettingRef points to cim22Setting or cim23Setting.'
  SUP top AUXILIARY
  MAY ( cimCollectionRef $ cimSettingRef )
)
```

3.20 cim23System

This class is a logical element that aggregates an enumerable set of managed system elements and operates as a functional whole. Within any particular subclass of system, there is a well-defined list of managed system element classes whose instances must be aggregated.

```
( 1.3.6.1.4.1.412.100.2.2.31 NAME 'cimNameFormat'
  DESC 'The System object and its derivatives are Top Level Objects
        of CIM. They provide the scope for numerous
        components. Having unique System keys is required. A
        heuristic can be defined in individual System subclasses to
        attempt to always generate the same System Name Key. The
        NameFormat property identifies how the System name was
        generated, using the subclass" heuristic.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.32 NAME 'cimPrimaryOwnerContact'
  DESC 'A string that provides information on how the primary
        system owner can be reached (e.g. phone number, email
        address, ...).'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.33 NAME 'cimPrimaryOwnerName'
  DESC 'The name of the primary system owner.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.34 NAME 'cimRoles'
```

```
  DESC 'An array (bag) of strings that specify the roles this
        System plays in the IT-environment. Subclasses of System
        may override this property to define explicit Roles
        values. Alternately, a Working Group may describe the
        heuristics, conventions and guidelines for specifying
        Roles. For example, for an instance of a networking system,
        the Roles property might contain the string, "Switch" or
        "Bridge".'
  SYNTAX DirectoryString
)

( 1.3.6.1.4.1.412.100.2.1.2.17 NAME 'cim23System'
  DESC 'A cim23System is a LogicalElement that aggregates an
        enumerable set of Managed System Elements. The aggregation
        operates as a functional whole. Within any particular
        subclass of System, there is a well-defined list of Managed
        System Element classes whose instances must be aggregated.'
  SUP cim23LogicalElement ABSTRACT
  MAY ( cimCreationClassName $ cimNameFormat $
       cimPrimaryOwnerContact $ cimPrimaryOwnerName $ cimRoles )
)
```

3.21 cim23ComputerSystem

This class is derived from cim23System and represents a special collection of managed system elements that provide compute capabilities. Thus, it serves as aggregation point to associate one or more of the following elements: file systems, operating systems, processors and memory (volatile and/or non-volatile storage).

```
( 1.3.6.1.4.1.412.100.2.2.35 NAME 'cimDedicated'
  DESC 'Enumeration indicating whether the ComputerSystem is a
        special-purpose System (ie, dedicated to a particular use),
        versus being "general purpose". For example, one could
        specify that the System is dedicated to "Print" (value=11)
        or acts as a "Hub" (value=8).   Values: 0="Not Dedicated",
        1="Unknown", 2="Other", 3="Storage", 4="Router",
        5="Switch", 6="Layer 3 Switch", 7="Central Office Switch",
        8="Hub", 9="Access Server", 10="Firewall", 11="Print",
        12="I/O", 13="Web Caching", 14="Management".'
  SYNTAX integer
)

( 1.3.6.1.4.1.412.100.2.1.2.18 NAME 'cim23ComputerSystem'
  DESC 'A class derived from System that is a special collection of
        ManagedSystemElements. This collection provides compute
        capabilities and serves as aggregation point to associate
        one or more of the following elements: FileSystem,
        OperatingSystem, Processor and Memory (Volatile and/or
        NonVolatile Storage).'
  SUP cim23System ABSTRACT
  MAY ( cimDedicated )
)
```

3.22 cim23LogicalDevice

This class represents an abstraction or emulation of a hardware entity, that may or may not be realized in physical hardware. Any characteristics of a logical device that are used to manage its operation or configuration are contained in, or associated with, this object.

```
( 1.3.6.1.4.1.412.100.2.2.36 NAME 'cimDeviceID'
  DESC 'An address or other identifying information to uniquely
        name the LogicalDevice.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.37 NAME 'cimPowerManagementSupported'
  DESC 'Boolean indicating that the Device can be power managed -
        ie, put into a power save state. This boolean does not
        show that power management features are currently
        enabled, or if enabled, what features are supported. Refer
        to the PowerManagementCapabilities array for this
        information. If this boolean is false, the integer value 1,
        for the string, "Not Supported", should be the only entry
        in the PowerManagementCapabilities array.'
  SYNTAX boolean SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.38 NAME 'cimPowerManagementCapabilities'
  DESC 'Shows the specific power-related capabilities of a
        LogicalDevice. The array values, 0="Unknown", 1="Not
        Supported" and 2="Disabled" are self-explanatory. The
        value, 3="Enabled" shows that the power management
        features are currently enabled but the exact feature set is
        unknown or the information is unavailable. "Power Saving
        Modes Entered Automatically" (4) describes that a Device
        can change its power state based on usage or other
        criteria. "Power State Settable" (5) shows that the
        SetPowerState method is supported. "Power Cycling
        Supported" (6) shows that the SetPowerState method can
        be invoked with the PowerState input variable set to 5
        ("Power Cycle"). "Timed Power On Supported" (7) shows
        that the SetPowerState method can be invoked with the
        PowerState input variable set to 5 ("Power Cycle") and the
        Time parameter set to a specific date and time, or
        interval, for power-on.   Values are 0="Unknown", 1="Not
        Supported", 2="Disabled", 3="Enabled", 4="Power Saving
        Modes Entered Automatically", 5="Power State Settable",
        6="Power Cycling Supported", 7="Timed Power On Supported".'
  SYNTAX integer
)

( 1.3.6.1.4.1.412.100.2.2.39 NAME 'cimAvailability'
  DESC 'The availability and status of the Device. For example, the
        Availability property shows that the Device is running
        and has full power (value=3), or is in a warning (4), test
        (5), degraded (10) or power save state (values 13-15 and
        17). Regarding the Power Save states, these are defined as
        follows: Value 13 ("Power Save - Unknown") shows that
        the Device is known to be in a power save mode, but its
```

```
        exact status in this mode is unknown; 14 ("Power Save - Low
        Power Mode") shows that the Device is in a power save
        state but still functioning, and may exhibit degraded
        performance; 15 ("Power Save - Standby") describes that the
        Device is not functioning but could be brought to full
        power "quickly"; and value 17 ("Power Save - Warning")
        shows that the Device is in a warning state, though
        also in a power save mode.   Values: 1="Other",
        2="Unknown", 3= "Running/Full Power", 4= "Warning", 5="In
        Test", 6="Not Applicable", 7= "Power Off", 8= "Off Line",
        9="Off Duty", 10="Degraded", 11= "Not Installed",
        12="Install Error", 13="Power Save - Unknown", 14="Power
        Save - Low Power Mode", 15="Power Save - Standby",
        16="Power Cycle", 17="Power Save - Warning", 18="Paused",
        19="Not Ready", 20="Not Configured".'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.40 NAME 'cimStatusInfo'
  DESC 'StatusInfo is a string indicating whether the LogicalDevice
        is in an enabled (value = 3), disabled (value = 4) or some
        other (1) or unknown (2) state. If this property does not
        apply to the LogicalDevice, the value, 5 ("Not
        Applicable"), should be used.  Values: 1="Other",
        2="Unknown", 3="Enabled", 4="Disabled", 5="Not Applicable"'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.41 NAME 'cimLastErrorCode'
  DESC 'LastErrorCode captures the last error code reported by the
        LogicalDevice.'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.42 NAME 'cimErrorDescription'
  DESC 'ErrorDescription is a free-form string supplying more
        information about the error recorded in LastErrorCode, and
        information on any corrective actions that may be taken.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.43 NAME 'cimErrorCleared'
  DESC 'ErrorCleared is a boolean property indicating that the
        error reported in LastErrorCode is now cleared.'
  SYNTAX boolean SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.70 NAME 'cimAdditionalAvailability'
  DESC 'Additional availability and status of the Device, beyond
        that specified in the Availability property. The
        Availability property denotes the primary status and
        availability of the Device. In some cases, this will not be
        sufficient to denote the complete status of the Device. In
        those cases, the AdditionalAvailability property can be used
        to provide further information. For example, a Device"s
        primary Availability may be "Off line" (value=8), but it may
        also be in a low power state (AdditonalAvailability
```

```
        value=14), or the Device could be running Diagnostics
        (AdditionalAvailability value=5, "In Test". Values
        1="Other", 2="Unknown", 3= "Running/Full Power",
        4="Warning", 5="In Test", 6="Not Applicable", 7="Power Off",
        8="Off Line", 9="Off Duty", 10="Degraded", 11="Not
        Installed", 12="Install Error", 13="Power Save - Unknown",
        14="Power Save - Low Power Mode", 15="Power Save - Standby",
        16="Power Cycle", 17="Power Save - Warning", 18="Paused",
        19="Not Ready", 20="Not Configured", 21="Quiesced"'
  SYNTAX integer
)

( 1.3.6.1.4.1.412.100.2.2.71 NAME 'cimMaxQuiesceTime'
  DESC 'Maximum time in milliseconds, that a Device can run in a
        "Quiesced" state. A Device"s state is defined in its
        Availability and AdditionalAvailability properties, where
        "Quiesced" is conveyed by the value 21. What occurs at the
        end of the time limit is device-specific. The Device may
        unquiesce, may offline or take other action. A value of 0
        indicates that a Device can remain quiesced indefinitely.'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.19 NAME 'cim23LogicalDevice'
  DESC 'An abstraction or emulation of a hardware entity, that may
        or may not be Realized in physical hardware. Any
        characteristics of a LogicalDevice that are used to manage
        its operation or configuration are contained in, or
        associated with, the LogicalDevice object. Examples of the
        operational properties of a Printer would be paper sizes
        supported, or detected errors. Examples of the
        configuration properties of a Sensor Device would be
        threshold settings. Various configurations could exist for
        a LogicalDevice. These configurations could be contained in
        Setting objects and associated with the LogicalDevice.'
  SUP cim23LogicalElement ABSTRACT
  MAY ( cimCreationClassName $ cimDeviceID $
       cimPowerManagementSupported $ cimAvailability $
       cimPowerManagementCapabilities $ cimStatusInfo $
       cimLastErrorCode $ cimErrorDescription $ cimErrorCleared $
       cimAdditionalAvailability $ cimMaxQuiesceTime )
)
```

3.23 cim23Service

This class represents a Logical Element that contains the information necessary to represent and manage the functionality provided by a device and/or software feature. A service is a general-purpose object to configure and manage the implementation of functionality. It is not the functionality itself.

```
( 1.3.6.1.4.1.412.100.2.2.44 NAME 'cimStartMode'
  DESC 'StartMode is a string value indicating whether the Service
        is automatically started by a System, Operating System,
        etc. or only started on request.  Values are "Automatic" and
        "Manual".'
  SYNTAX DirectoryString{10} SINGLE-VALUE
)
```

```
( 1.3.6.1.4.1.412.100.2.2.45 NAME 'cimStarted'
  DESC 'Started is a boolean indicating whether the Service has
        been started (TRUE), or stopped (FALSE).'
  SYNTAX boolean SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.20 NAME 'cim23Service'
  DESC 'A cim23Service is a Logical Element that contains the
        information necessary to represent and manage the
        functionality provided by a Device and/or
        SoftwareFeature. A Service is a general-purpose object to
        configure and manage the implementation of
        functionality. It is not the functionality itself.'
  SUP cim23LogicalElement ABSTRACT
  MAY ( cimCreationClassName $ cimStartMode $ cimStarted )
)
```

3.24 cim23ServiceAccessPoint

This class represents the ability to use or invoke a service. Access points represent that a service is made available to other entities for use.

```
( 1.3.6.1.4.1.412.100.2.1.2.21 NAME 'cim23ServiceAccessPoint'
  DESC 'cim23ServiceAccessPoint represents the ability to use or
        invoke a Service. Access points represent that a Service is
        made available to other entities for use.'
  SUP cim23LogicalElement ABSTRACT
  MAY ( cimCreationClassName )
)
```

3.25 cim22DependencyAuxClass

This class represents a generic association used to establish dependency relationships between objects.

```
( 1.3.6.1.4.1.412.100.2.2.46 NAME 'cimAntecedentRef'
  DESC 'Antecedent represents the independent object in this
        association.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.47 NAME 'cimDependentRef'
  DESC 'Dependent represents the object dependent on the
        Antecedent.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.22 NAME 'cim22DependencyAuxClass'
  DESC 'cimDependency is a generic association used to establish
        dependency relationships between objects. Both attributes
        point to cim22ManagedSystemElement or
        cim23ManagedSystemElement objects.'
  SUP top AUXILIARY
  MAY ( cimAntecedentRef $ cimDependentRef )
```

```
)
```

3.26 cim22ServiceAccessBySAPAuxClass

This association identifies the access points for a service. For example, a printer may be accessed by Netware, MacIntosh or Windows service access points, potentially hosted on different systems.

```
( 1.3.6.1.4.1.412.100.2.1.1.23
  NAME 'cim22ServiceAccessBySAPAuxClass'
  DESC 'ServiceAccessBySAP is an association that identifies
        the access points for a Service. For example, a printer may
        be accessed by Netware, MacIntosh or Windows
        ServiceAccessPoints, potentially hosted on different
        Systems. Attribute cimAntecedentRef points to cim22Service
        or cim23Service and attribute cimDependentRef points to
        cim22ServiceAccessPoint or cim23ServiceAccessPoint.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.27 cim23ServiceServiceDependencyAuxClass

This is an association between two services, showing that the latter is required to be present, required to have completed, or must be absent for the former Service to provide its functionality. For example, boot Services may be dependent on underlying BIOS disk and initialization services. For initialization services, the boot service is simply dependent on the initialization services completing.

```
( 1.3.6.1.4.1.412.100.2.2.48 NAME 'cimTypeOfDependency'
  DESC 'The Service to Service dependency. This property describes
        that the associated Service must have completed (value=2),
        must be started (3) or must not be started (4) in order for
        the Service to function.   Values are 0="Unknown",
        1="Other", 2="Service Must Have Completed", 3="Service Must
        Be Started", 4="Service Must Not Be Started".'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.72 NAME 'cimRestartService'
  DESC 'this property describes that the antecedent service must be
        restarted after the dependent operation is complete.'
  SYNTAX boolean SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.24
  NAME 'cim23ServiceServiceDependencyAuxClass'
  DESC 'ServiceServiceDependency is an association between a
        Service and another Service, indicating that the latter is
        required to be present, required to have completed, or must
        be absent for the former Service to provide its
        functionality. For example, Boot Services may be dependent
        on underlying BIOS Disk and initialization Services. In
        the case of the initialization Services, the Boot Service
        is simply dependent on the init Services completing. For
        the Disk Services, Boot Services may actually use the
```

```
        SAPs of this Service. This usage dependency is modeled via
        the ServiceSAPDependency association. Both attributes
        point to cim23Service objects.'
  SUP cim22DependencyAuxClass AUXILIARY
  MAY ( cimRestartService $ cimTypeOfDependency )
)
```

3.28 cim22ServiceSAPDependencyAuxClass

This class is an association between a service and a service access point showing that the referenced SAP is used by the service to provide its functionality. For example, boot services may invoke BIOS disk services (interrupts) to function.

```
( 1.3.6.1.4.1.412.100.2.1.1.25
  NAME 'cim22ServiceSAPDependencyAuxClass'
  DESC 'ServiceSAPDependency is an association between a
        Service and a ServiceAccessPoint indicating that the
        referenced SAP is used by the Service to provide its
        functionality. For example, Boot Services may invoke BIOS"
        Disk Services (interrupts) to function. Attribute
        cimAntecedentRef points to cim22ServiceAccessPoint or
        cim23ServiceAccessPoint and attribute cimDependentRef
        points to cim22Service or cim23Service.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.29 cim22SAPSAPDependencyAuxClass

This class is an association between two service access points showing that the latter is required in order for the former to use or connect with its service. For example, to print at a network printer, local print access points must use underlying network-related SAPs, or protocol endpoints, to send the print request.

```
( 1.3.6.1.4.1.412.100.2.1.1.26 NAME 'cim22SAPSAPDependencyAuxClass'
  DESC 'SAPSAPDependency is an association between a
        ServiceAccessPoint and another ServiceAccessPoint
        indicating that the latter is required in order for the
        former ServiceAccessPoint to use or connect with its
        Service. For example, to print at a network printer, local
        Print Access Points must use underlying network-related
        SAPs, or ProtocolEndpoints, to send the print request.
        Both attributes point to cim22ServiceAccessPoint or
        cim23ServiceAccessPoint objects.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.30 cim22ProvidesServiceToElementAuxClass

This association is used to describe that ManagedSystemElements may be dependent on the functionality of one or more Services.

```
( 1.3.6.1.4.1.412.100.2.1.1.27
  NAME 'cim22ProvidesServiceToElementAuxClass'
  DESC 'ProvidesServiceToElement is used to describe that
        ManagedSystemElements may be dependent on the functionality
        of one or more Services. An example is that a Processor and
        an Enclosure (PhysicalElement) are dependent on AlertOnLAN
```

```
        Services to signal an incomplete or erroneous boot, and
        hardware-related errors. Attribute cimAntecedentRef points
        to cim22Service or cim23Service and attribute
        cimDependentRef points to cim22ManagedSystemElement or
        cim23ManagedSystemElement.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.31 cim22RealizesAuxClass

This association defines the mapping between a logical device and the physical component that implements the device.

```
( 1.3.6.1.4.1.412.100.2.1.1.28 NAME 'cim22RealizesAuxClass'
  DESC 'Realizes is the association that defines the mapping
      between a Logical Device and the physical component that
      implements the Device. Attribute cimAntecedentRef points to
      cim22PhysicalElement or cim23PhysicalElement and attribute
      cimDependentRef points to cim22LogicalDevice or
      cim23LogicalDevice.'
  SUP cim22DependencyAuxClass AUXILIARY
)
```

3.32 cim22ComponentAuxClass

This class is a generic association used to establish 'part of' relationships between managed system elements. For example, the system component association defines parts of a system.

```
( 1.3.6.1.4.1.412.100.2.2.49 NAME 'cimGroupComponentRef'
  DESC 'The parent element in the association.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.50 NAME 'cimPartComponentRef'
  DESC 'The child element in the association.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.29 NAME 'cim22ComponentAuxClass'
  DESC 'Component is a generic association used to establish
        "part of" relationships between Managed System
        Elements. For example, the SystemComponent association
        defines parts of a System. Both attributes point to
        cim22ManagedSystemElement or cim23ManagedSystemElement
        objects.'
  SUP top AUXILIARY
  MAY ( cimGroupComponentRef $ cimPartComponentRef )
)
```

3.33 cim22SystemComponentAuxClass

This class is a specialization of the cim22ComponentAuxClass association that establishes part of relationships between a system and the managed system elements of which it is composed.

```
( 1.3.6.1.4.1.412.100.2.1.1.30 NAME 'cim22SystemComponentAuxClass'
```

```
  DESC 'SystemComponent is a specialization of the
        Component association that establishes "part of"
        relationships between a System and the Managed System
        Elements of which it is composed. Attribute
        cimGroupComponentRef points to cim22System or cim23System
        and attribute cimPartComponentRef points to
        cim22ManagedSystemElement or cim23ManagedSystemElement.'
  SUP cim22ComponentAuxClass AUXILIARY
)
```

3.34 cim22ServiceComponentAuxClass

This auxiliary class models a set of subordinate services that are aggregated together to form a higher-level service.

```
( 1.3.6.1.4.1.412.100.2.1.1.31 NAME 'cim22ServiceComponentAuxClass'
  DESC 'The ServiceComponent aggregation models a set of
        subordinate Services that are aggregated together to form a
        higher-level service. Both attributes point to cim22Service
        or cim23Service objects.'
  SUP cim22ComponentAuxClass AUXILIARY
)
```

3.35 cim23Product

This concrete class that is a collection of physical elements, software features and/or other products, acquired as a unit. Acquisition implies an agreement between supplier and consumer that may have implications to product licensing, support and warranty.

```
( 1.3.6.1.4.1.412.100.2.2.51 NAME 'cimIdentifyingNumber'
  DESC 'Product identification such as a serial number on software,
        a die number on a hardware chip, or (for non-commercial
        Products) a project number.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.52 NAME 'cimVendor'
  DESC 'The name of the Product"s supplier, or entity selling the
        Product (the manufacturer, reseller, OEM, etc.).
        Corresponds to the Vendor property in the Product object in
        the CIM Solution Exchange Standard.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.53 NAME 'cimSKUNumber'
  DESC 'Product SKU (stock keeping unit) information.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.73 NAME 'cimWarrantyStartDate'
  DESC 'If this Product is under warranty, the start date
        of the warranty.'
  SYNTAX generalizedTime SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.74 NAME 'cimWarrantyDuration'
  DESC 'If this Product is under warranty, the duration of the
```

```
        warranty in days.'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.32 NAME 'cim23Product'
  DESC 'cim23Product is a concrete class that is a collection of
        PhysicalElements, SoftwareFeatures and/or other Products,
        acquired as a unit. Acquisition implies an agreement
        between supplier and consumer that may have implications
        to Product licensing, support and warranty. Non-commercial
        (e.g., in-house developed Products) should also be
        identified as an instance of cim23Product.'
  SUP cim23ManagedElement
  MAY ( cimIdentifyingNumber $ cimName $ cimSKUNumber $ cimVendor $
        cimVersion $ cimWarrantyStartDate $ cimWarrantyDuration )
)

( 1.3.6.1.4.1.412.100.2.3.2.2 NAME 'cim23ProductNameForm1'
  OC cim23Product
  MUST ( orderedCimKeys )
)

( <sr2> NAME 'cim23ProductStructureRule1'
  FORM cim23ProductNameForm1
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23Product.

```
( 1.3.6.1.4.1.412.100.2.1.2.32 NAME 'cim23ProductContentRule'
  DESC 'The auxiliary classes that may be attached to cim23Product'
  AUX ( cim22ProductParentChildAuxClass $
       cim22CompatibleProductAuxClass $
       cim22ProductProductDependencyAuxClass $
       cim22ProductSupportAuxClass $ cim22ProductFRUAuxClass $
       cim22ProductPhysicalElementsAuxClass $
       cim22FRUIncludesProductAuxClass $
       cim22DependencyAuxClass $ cim23MemberOfCollectionAuxClass )
)
```

3.36 cim22ProductParentChildAuxClass

The association defines a parent child hierarchy among products. For example, a product may come bundled with other products.

```
( 1.3.6.1.4.1.412.100.2.2.54 NAME 'cimParentRef'
  DESC 'The parent Product in the association.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.55 NAME 'cimChildRef'
  DESC 'The child Product in the association.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.33
```

```
  NAME 'cim22ProductParentChildAuxClass'
  DESC 'The ProductParentChild association defines a parent
        child hierarchy among Products. For example, a Product may
        come bundled with other Products. Both attributes point to
        cim22Product or cim23Product objects.'
  SUP top AUXILIARY
  MAY ( cimParentRef $ cimChildRef )
)
```

3.37 cim22CompatibleProductAuxClass

This association between products can show a wide variety of information. For example, it can show that the two referenced products interoperate, that they can be installed together, that one can be the physical container for the other, etc.

```
( 1.3.6.1.4.1.412.100.2.2.56 NAME 'cimProductRef'
  DESC 'The Product for which compatible offerings are defined.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.57 NAME 'cimCompatibleProductRef'
  DESC 'The compatible Product.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.58 NAME 'cimCompatibilityDescription'
  DESC 'CompatibilityDescription is a free-form string defining
        how the two referenced Products interoperate or are
        compatible, any limitations to compatibility, etc.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.1.34 NAME 'cim22CompatibleProductAuxClass'
  DESC 'CompatibleProduct is an association between Products
        that can show a wide variety of information. For
        example, it can show that the two referenced Products
        interoperate, that they can be installed together, that
        one can be the physical container for the other, etc. The
        string property, CompatibilityDescription, defines how the
        Products interoperate or are compatible, any limitations
        regarding interoperability or installation, ...  Both
        reference attributes point to cim22Product or cim23Product
        objects.'
  SUP top AUXILIARY
  MAY ( cimProductRef $ cimCompatibleProductRef $
       cimCompatibilityDescription )
)
```

3.38 cim22ProductProductDependencyAuxClass

This association is between two products, showing that one must be installed, or must be absent, for the other to function. This is conceptually equivalent to the service to service dependency association.

```
( 1.3.6.1.4.1.412.100.2.2.59 NAME 'cimRequiredProductRef'
  DESC 'The required Product.'
```

```
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.60 NAME 'cimDependentProductRef'
  DESC 'The Product that is dependent on another Product.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.35
  NAME 'cim22ProductProductDependencyAuxClass'
  DESC 'ProductProductDependency is an association between two
        Products, indicating that one must be installed, or must
        be absent, for the other to function. This is conceptually
        equivalent to the ServiceServiceDependency association.
        Both reference attributes point to cim22Product or
        cim23Product objects.'
  SUP top AUXILIARY
  MAY ( cimRequiredProductRef $ cimDependentProductRef $
       cimTypeOfDependency )
)
```

3.39 cim23SupportAccess

This class defines how to obtain help for a product.

```
( 1.3.6.1.4.1.412.100.2.2.61 NAME 'cimSupportAccessId'
  DESC 'SupportAccessID is an arbitrary, free form string defined
        by the Product Vendor or by the organization that deploys
        the Product. This property, since it is a key, should be
        unique throughout the enterprise.'
  SYNTAX DirectoryString{256} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.62 NAME 'cimCommunicationInfo'
  DESC 'CommunicationInfo provides the details of the
        CommunicationMode. For example, if the CommunicationMode is
        "Phone", CommunicationInfo specifies the phone number to be
        called.'
  SYNTAX DirectoryString SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.63 NAME 'cimCommunicationMode'
  DESC 'CommunicationMode defines the form of communication in
        order to obtain support. For example, phone communication
        (value=2), fax (3) or email (8) can be specified.  Values:
        1="Other", 2="Phone", 3="Fax", 4="BBS", 5="Online Service",
        6="Web Page", 7="FTP", 8="E-mail"'
  SYNTAX integer SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.64 NAME 'cimLocale'
  DESC 'Locale defines the geographic region and/or language
        dialect to which this Support resource pertains.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.36 NAME 'cim23SupportAccess'
```

```
  DESC 'The cim23SupportAccess association defines how to obtain
        help for a Product.'
  SUP cim23ManagedElement
  MAY ( cimCommunicationInfo $ cimCommunicationMode $
       cimDescription $ cimLocale $ cimSupportAccessId )
)

( 1.3.6.1.4.1.412.100.2.3.2.3 NAME 'cim23SupportAccessNameForm1'
  OC cim23SupportAccess
  MUST ( orderedCimKeys )
)

( <sr3> NAME 'cim23SupportAccessStructureRule1'
  FORM cim23SupportAccessNameForm1
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23SupportAccessProduct.

```
( 1.3.6.1.4.1.412.100.2.1.2.36 NAME 'cim23SupportAccessContentRule'
  DESC 'The auxiliary classes that may be attached to
        cim23SupportAccess'
  AUX ( cim22ProductSupportAuxClass $
       cim22DependencyAuxClass $ cim23MemberOfCollectionAuxClass )
)
```

3.40 cim22ProductSupportAuxClass

This auxiliary class represents the association between products and support access that conveys how support is obtained for the product. This is a many-to-many relationship, implying that various types of support are available for a product, and that the same support object can provide help for multiple products. This class defines two attributes that are self-explanatory.

```
( 1.3.6.1.4.1.412.100.2.2.65 NAME 'cimSupportRef'
  DESC 'Support for the Product.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.37 NAME 'cim22ProductSupportAuxClass'
  DESC 'cimProductSupport is an association between Product and
      SupportAccess that conveys how support is obtained for the
      Product. This is a many-to-many relationship, implying that
      various types of Support are available for a Product, and
      that the same Support object can provide help for
      multiple Products. Attribute cimProductRef points to
      cim22Product or cim23Product and attribute cimSupportRef
      points to cim22SupportAccess or cim23SupportAccess.'
  SUP top AUXILIARY
  MAY ( cimProductRef $ cimSupportRef )
)
```

3.41 cim23FRU

This class is a vendor-defined collection of products and/or physical elements that is associated with a product for supporting, maintaining or upgrading that product at the customer's location. FRU is an acronym for 'field replaceable unit'.

```
( 1.3.6.1.4.1.412.100.2.2.66 NAME 'cimFRUNumber'
  DESC 'FRU ordering information.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.67 NAME 'cimRevisionLevel'
  DESC 'The FRU"s revision level.'
  SYNTAX DirectoryString{64} SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.38 NAME 'cim23FRU'
  DESC 'The cimFRU class is a vendor-defined collection of
        Products and/or PhysicalElements that is associated with a
        Product for supporting, maintaining or upgrading that
        Product at the customer\'s location. FRU is an acronym for
        "field replaceable unit". '
  SUP cim23ManagedElement
  MAY ( cimFRUNumber $ cimIdentifyingNumber $ cimName $ cimVendor $
        cimRevisionLevel )
)

( 1.3.6.1.4.1.412.100.2.3.2.4 NAME 'cim23FRUNameForm1'
  OC cim23FRU
  MUST ( orderedCimKeys )
)

( <sr4> NAME 'cim23FRUStructureRule1'
  FORM cim23FRUNameForm1
)
```

The following content rule specifies the auxiliary classes that may be attached to cim23FRU.

```
( 1.3.6.1.4.1.412.100.2.1.2.38 NAME 'cim23FRUContentRule'
  DESC 'The auxiliary classes that may be attached to cim23FRU'
  AUX ( cim22ProductFRUAuxClass $ cim22FRUPhysicalElementsAuxClass $
        cim23FRUIncludesProductAuxClass $
        cim22DependencyAuxClass $ cim23MemberOfCollectionAuxClass )
)
```

3.42 cim22ProductFRUAuxClass

This auxiliary class provides information regarding what product components have been or are being replaced and uses the previously defined attribute cimProductRefs.

```
( 1.3.6.1.4.1.412.100.2.2.68 NAME 'cimFRURef'
  DESC 'The FRU.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.39 NAME 'cim22ProductFRUAuxClass'
  DESC 'cimProductFRU is an association between Product and FRU
        that provides information regarding what Product components
        have been or are being replaced. The association is one to
        many, conveying that a Product can have many FRUs, and that
        a particular instance of a FRU is only applied to one
        (instance of a) Product. Attribute cimProductRef points to
```

```
        cim22Product or cim23Product and attribute cimFRURef points
        to cim22FRU or cim23FRU.'
  SUP top AUXILIARY
  MAY ( cimProductRef $ cimFRURef )
)
```

3.43 cim22ProductPhysicalElementsAuxClass

Shows the physical elements that make up a product.

```
( 1.3.6.1.4.1.412.100.2.2.22 NAME 'cimComponentRef'
  DESC 'The PhysicalElement that is a part of the Product.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.1.1.40
  NAME 'cim22ProductPhysicalElementsAuxClass'
  DESC 'Shows the PhysicalElements that make up a
        Product. Attribute cimProductRef points to
        cim22Product or cim23Product and attribute cimComponentRef
        points to cim22PhysicalElement or cim23PhysicalElement.'
  SUP top AUXILIARY
  MAY ( cimProductRef $ cimComponentRef )
)
```

3.44 cim22FRUPhysicalElementsAuxClass

This auxiliary class shows the physical elements that make up a FRU.

```
( 1.3.6.1.4.1.412.100.2.1.1.41
  NAME 'cim22FRUPhysicalElementsAuxClass'
  DESC 'Shows the PhysicalElements that make up a FRU. Attribute
        cimFRURef points to cim22FRU or cim23FRU and attribute
        cimComponentRef points to cim22PhysicalElement or
        cim23PhysicalElement.'
  SUP top AUXILIARY
  MAY ( cimFRURef $ cimComponentRef )
)
```

3.45 cim22FRUIncludesProductAuxClass

This auxiliary class shows that a FRU may be composed of other product(s).

```
( 1.3.6.1.4.1.412.100.2.1.1.42
  NAME 'cim22FRUIncludesProductAuxClass'
  DESC 'Shows that a FRU may be composed of other
      Product(s). Attribute cimFRURef points to cim22FRU or cim23FRU
      and attribute cimComponentRef points to cim22Product or
      cim23Product.'
  SUP top AUXILIARY
  MAY ( cimFRURef $ cimComponentRef )
)
```

3.46 cim23SynchronizedAuxClass

This auxiliary class indicates that two logical elements were aligned or made to be equivalent at the specified point in time. Preservation of synchronization is determined by the value of the cimSyncMaintained attribute.

```
( 1.3.6.1.4.1.412.100.2.2.75 NAME 'cimSyncedElementRef'
  DESC 'SyncedElement represents another LogicalElement that is
        synchronized with the entity referenced as SystemElement.'
  SYNTAX DN
)

( 1.3.6.1.4.1.412.100.2.2.76 NAME 'cimWhenSynced'
  DESC 'The point in time that the Elements were synchronized.'
  SYNTAX generalizedTime SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.2.77 NAME 'cimSyncMaintained'
  DESC 'Boolean indicating whether synchronization is maintained.'
  SYNTAX boolean SINGLE-VALUE
)

( 1.3.6.1.4.1.412.100.2.1.2.47 NAME 'cim23SynchronizedAuxClass'
  DESC 'Indicates that two LogicalElements were aligned or made to
        be equivalent at the specified point in time. If the
        boolean property SyncMaintained is TRUE, then
        synchronization of the Elements is preserved. Both like and
        unlike objects may be synchronized. For example, two
        WatchDog timers may be aligned, or the contents of a
        LogicalFile may be synchronized with the contents of a
        StorageExtent. Both attributes point to cim23LogicalElement
        objects.'
  SUP top AUXILIARY
  MAY ( cimSystemElementRef $ cimSyncedElementRef $ cimWhenSynced $
       cimSyncMaintained )
)
```

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, December 1997.

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

[3] CIM, "CIM Core Model, v2.3," http://www.dmtf.org/spec/cim_schema_v23.html.

[4] F. Yergeau, "UTF-8, a transformation format of ISO 10646," RFC 2279, January 1998.

5. Acknowledgment

This work is a product of the DMTF LDAP Mapping Working Group and has benefited from many comments and discussions during this groups meetings.

A. Structural Rules

The following table states the structural rules defined in this document.

Rule	Structural Class	RDN Attribute	Superior Rules	Defined
<sr1>	cim23Configuration	orderedCimKeys	*	3.11
<sr2>	cim23Product	orderedCimKeys	*	3.35
<sr3>	cim23SupportAccess	orderedCimKeys	*	3.39
<sr4>	cim23FRU	orderedCimKeys	*	3.41
<sr5>	cimAssociationInstance	cimAssociationName	<sr1>, <sr2>, <sr3>, <sr4>	2.3.1
<sr6>	cimOtherIdentifyingInfoInstance	arrayIndex	**	2.3.2

* 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.

** The superiors for this rule are not defined in this mapping. In subsequent DMTF CIM mapping documents which define mappings of non-abstract subclasses of CIM_ComputerSystem and CIM_LogicalDevice, it will be possible to define the possible superiors for cimOtherIdentifyingInfoInstance.

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
1.3.6.1.4.1.412.100.1.1.1.1	cimAssociationInstance	2.3.2
1.3.6.1.4.1.412.100.2.1.2.1	cim23ManagedSystemElement	3.2
1.3.6.1.4.1.412.100.2.1.2.2	cim23CollectionOfMSEs	3.4
1.3.6.1.4.1.412.100.2.1.2.3	cim23CollectedMSEsAuxClass	3.6
1.3.6.1.4.1.412.100.2.1.1.4	cim22CollectedCollectionsAuxClass	3.7
1.3.6.1.4.1.412.100.2.1.2.5	cim23PhysicalElement	3.8
1.3.6.1.4.1.412.100.2.1.2.6	cim22LogicalElement	3.9
1.3.6.1.4.1.412.100.2.1.1.7	cim22LogicalIdentityAuxClass	3.10
1.3.6.1.4.1.412.100.2.1.2.8	cim23Configuration	3.11
1.3.6.1.4.1.412.100.2.1.1.9	cim22ConfigurationComponentAuxClass	3.12
1.3.6.1.4.1.412.100.2.1.1.10	cim22ElementConfigurationAuxClass	3.13
1.3.6.1.4.1.412.100.2.1.1.11	cim22CollectionConfigurationAuxClass	3.14

1.3.6.1.4.1.412.100.2.1.2.12	cim23Setting	3.15
1.3.6.1.4.1.412.100.2.1.1.13	cim22ElementSettingAuxClass	3.16
1.3.6.1.4.1.412.100.2.1.1.14	cim22DefaultSettingAuxClass	3.17
1.3.6.1.4.1.412.100.2.1.1.15	cim22SettingContextAuxClass	3.18
1.3.6.1.4.1.412.100.2.1.1.16	cim22CollectionSettingAuxClass	3.19
1.3.6.1.4.1.412.100.2.1.2.17	cim23System	3.20
1.3.6.1.4.1.412.100.2.1.2.18	cim23ComputerSystem	3.21
1.3.6.1.4.1.412.100.2.1.2.19	cim23LogicalDevice	3.22
1.3.6.1.4.1.412.100.2.1.2.20	cim23Service	3.23
1.3.6.1.4.1.412.100.2.1.2.21	cim23ServiceAccessPoint	3.24
1.3.6.1.4.1.412.100.2.1.1.22	cim22DependencyAuxClass	3.25
1.3.6.1.4.1.412.100.2.1.1.23	cim22ServiceAccessBySAPAuxClass	3.26
1.3.6.1.4.1.412.100.2.1.2.24	cim23ServiceServiceDependencyAuxClass	3.27
1.3.6.1.4.1.412.100.2.1.1.25	cim22ServiceSAPDependencyAuxClass	3.28
1.3.6.1.4.1.412.100.2.1.1.26	cim22SAPSAPDependencyAuxClass	3.29
1.3.6.1.4.1.412.100.2.1.1.27	cim22ProvidesServiceToElementAuxClass	3.30
1.3.6.1.4.1.412.100.2.1.1.28	cim22RealizesAuxClass	3.31
1.3.6.1.4.1.412.100.2.1.1.29	cim22ComponentAuxClass	3.32
1.3.6.1.4.1.412.100.2.1.1.30	cim22SystemComponentAuxClass	3.33
1.3.6.1.4.1.412.100.2.1.1.31	cim22ServiceComponentAuxClass	3.34
1.3.6.1.4.1.412.100.2.1.2.32	cim23Product	3.35
1.3.6.1.4.1.412.100.2.1.1.33	cim22ProductParentChildAuxClass	3.36
1.3.6.1.4.1.412.100.2.1.1.34	cim22CompatibleProductAuxClass	3.37
1.3.6.1.4.1.412.100.2.1.1.35	cim22ProductProductDependencyAuxClass	3.38
1.3.6.1.4.1.412.100.2.1.2.36	cim23SupportAccess	3.39
1.3.6.1.4.1.412.100.2.1.1.37	cim22ProductSupportAuxClass	3.40
1.3.6.1.4.1.412.100.2.1.2.38	cim23FRU	3.41
1.3.6.1.4.1.412.100.2.1.1.39	cim22ProductFRUAuxClass	3.42
1.3.6.1.4.1.412.100.2.1.1.40	cim22ProductPhysicalElementsAuxClass	3.43
1.3.6.1.4.1.412.100.2.1.1.41	cim22FRUPhysicalElementsAuxClass	3.44
1.3.6.1.4.1.412.100.2.1.1.42	cim22FRUIncludesProductAuxClass	3.45
1.3.6.1.4.1.412.100.2.1.1.43	cimOtherIdentifyingInfoInstance	2.3
1.3.6.1.4.1.412.100.2.1.2.44	cim23ManagedElement	3.1
1.3.6.1.4.1.412.100.2.1.2.45	cim23Collection	3.3
1.3.6.1.4.1.412.100.2.1.2.46	cim23MemberOfCollectionAuxClass	3.5
1.3.6.1.4.1.412.100.2.1.2.47	cim23SynchronizedAuxClass	3.46

B.2 Attributes

OID	Attribute	Section
1.3.6.1.4.1.412.100.1.2.1	orderedCimKeys	2.4

1.3.6.1.4.1.412.100.1.2.2	orderedCimModelPath	2.4
1.3.6.1.4.1.412.100.1.2.3	cimAssociationName	2.3.1
1.3.6.1.4.1.412.100.1.2.4	cimAssociationTypeName	2.3.1
1.3.6.1.4.1.412.100.1.2.5	arrayIndex	2.3.2
1.3.6.1.4.1.412.100.2.2.1	cimCaption	3.1
1.3.6.1.4.1.412.100.2.2.2	cimDescription	3.1
1.3.6.1.4.1.412.100.2.2.3	cimInstallDate	3.2
1.3.6.1.4.1.412.100.2.2.4	cimName	3.2
1.3.6.1.4.1.412.100.2.2.5	cimStatus	3.2
1.3.6.1.4.1.412.100.2.2.6	cimCollectionID	3.4
1.3.6.1.4.1.412.100.2.2.7	cimCollectionRef	3.5
1.3.6.1.4.1.412.100.2.2.8	cimMemberRef	3.5
1.3.6.1.4.1.412.100.2.2.9	cimCollectionInCollectionRef	3.7
1.3.6.1.4.1.412.100.2.2.10	cimCreationClassName	3.8
1.3.6.1.4.1.412.100.2.2.11	cimTag	3.8
1.3.6.1.4.1.412.100.2.2.12	cimManufacturer	3.8
1.3.6.1.4.1.412.100.2.2.13	cimModel	3.8
1.3.6.1.4.1.412.100.2.2.14	cimSKU	3.8
1.3.6.1.4.1.412.100.2.2.15	cimSerialNumber	3.8
1.3.6.1.4.1.412.100.2.2.16	cimVersion	3.8
1.3.6.1.4.1.412.100.2.2.17	cimPartNumber	3.8
1.3.6.1.4.1.412.100.2.2.18	cimOtherIdentifyingInfo	3.8
1.3.6.1.4.1.412.100.2.2.19	cimPoweredOn	3.8
1.3.6.1.4.1.412.100.2.2.20	cimSystemElementRef	3.10
1.3.6.1.4.1.412.100.2.2.21	cimSameElementRef	3.10
1.3.6.1.4.1.412.100.2.2.22	cimComponentRef	3.43
1.3.6.1.4.1.412.100.2.2.23	cimIdentifyingDescription	2.3.2
1.3.6.1.4.1.412.100.2.2.24	cimConfigGroupRef	3.12
1.3.6.1.4.1.412.100.2.2.25	cimConfigComponentRef	3.12
1.3.6.1.4.1.412.100.2.2.26	cimElementRef	3.13
1.3.6.1.4.1.412.100.2.2.27	cimConfigurationRef	3.13
1.3.6.1.4.1.412.100.2.2.28	cimSettingID	3.15
1.3.6.1.4.1.412.100.2.2.29	cimSettingRef	3.16
1.3.6.1.4.1.412.100.2.2.30	cimContextRef	3.18
1.3.6.1.4.1.412.100.2.2.31	cimNameFormat	3.20
1.3.6.1.4.1.412.100.2.2.32	cimPrimaryOwnerContact	3.20
1.3.6.1.4.1.412.100.2.2.33	cimPrimaryOwnerName	3.20
1.3.6.1.4.1.412.100.2.2.34	cimRoles	3.20
1.3.6.1.4.1.412.100.2.2.35	cimDedicated	3.21
1.3.6.1.4.1.412.100.2.2.36	cimDeviceID	3.22

1.3.6.1.4.1.412.100.2.2.37	cimPowerManagementSupported	3.22
1.3.6.1.4.1.412.100.2.2.38	cimPowerManagementCapabilities	3.22
1.3.6.1.4.1.412.100.2.2.39	cimAvailability	3.22
1.3.6.1.4.1.412.100.2.2.40	cimStatusInfo	3.22
1.3.6.1.4.1.412.100.2.2.41	cimLastErrorCode	3.22
1.3.6.1.4.1.412.100.2.2.42	cimErrorDescription	3.22
1.3.6.1.4.1.412.100.2.2.43	cimErrorCleared	3.22
1.3.6.1.4.1.412.100.2.2.44	cimStartMode	3.23
1.3.6.1.4.1.412.100.2.2.45	cimStarted	3.23
1.3.6.1.4.1.412.100.2.2.46	cimAntecedentRef	3.25
1.3.6.1.4.1.412.100.2.2.47	cimDependentRef	3.25
1.3.6.1.4.1.412.100.2.2.48	cimTypeOfDependency	3.27
1.3.6.1.4.1.412.100.2.2.49	cimGroupComponentRef	3.32
1.3.6.1.4.1.412.100.2.2.50	cimPartComponentRef	3.32
1.3.6.1.4.1.412.100.2.2.51	cimIdentifyingNumber	3.35
1.3.6.1.4.1.412.100.2.2.52	cimVendor	3.35
1.3.6.1.4.1.412.100.2.2.53	cimSKUNumber	3.35
1.3.6.1.4.1.412.100.2.2.54	cimParentRef	3.36
1.3.6.1.4.1.412.100.2.2.55	cimChildRef	3.36
1.3.6.1.4.1.412.100.2.2.56	cimProductRef	3.37
1.3.6.1.4.1.412.100.2.2.57	cimCompatibleProductRef	3.37
1.3.6.1.4.1.412.100.2.2.58	cimCompatibilityDescription	3.37
1.3.6.1.4.1.412.100.2.2.59	cimRequiredProductRef	3.38
1.3.6.1.4.1.412.100.2.2.60	cimDependentProductRef	3.38
1.3.6.1.4.1.412.100.2.2.61	cimSupportAccessId	3.39
1.3.6.1.4.1.412.100.2.2.62	cimCommunicationInfo	3.39
1.3.6.1.4.1.412.100.2.2.63	cimCommunicationMode	3.39
1.3.6.1.4.1.412.100.2.2.64	cimLocale	3.39
1.3.6.1.4.1.412.100.2.2.65	cimSupportRef	3.40
1.3.6.1.4.1.412.100.2.2.66	cimFRUNumber	3.41
1.3.6.1.4.1.412.100.2.2.67	cimRevisionLevel	3.41
1.3.6.1.4.1.412.100.2.2.68	cimFRURef	3.42
1.3.6.1.4.1.412.100.2.2.69	cimManufactureDate	3.8
1.3.6.1.4.1.412.100.2.2.70	cimAdditionalAvailability	3.22
1.3.6.1.4.1.412.100.2.2.71	cimMaxQuiesceTime	3.22
1.3.6.1.4.1.412.100.2.2.72	cimRestartService	3.27
1.3.6.1.4.1.412.100.2.2.73	cimWarrantyStartDate	3.35
1.3.6.1.4.1.412.100.2.2.74	cimWarrantyDuration	3.35
1.3.6.1.4.1.412.100.2.2.75	cimSyncedElementRef	3.46
1.3.6.1.4.1.412.100.2.2.76	cimWhenSynced	3.46

1.3.6.1.4.1.412.100.2.2.77	cimSyncMaintained	3.46

B.3 Nameforms

OID	Nameform	Section
1.3.6.1.4.1.412.100.1.3.1.1	cimAssociationInstanceNameForm	2.3.1
1.3.6.1.4.1.412.100.2.3.2.1	cim23ConfigurationNameForm1	3.11
1.3.6.1.4.1.412.100.2.3.2.2	cim23ProductNameForm1	3.35
1.3.6.1.4.1.412.100.2.3.2.3	cim23SupportAccessNameForm1	3.39
1.3.6.1.4.1.412.100.2.3.2.4	cim23FRUNameForm1	3.41
1.3.6.1.4.1.412.100.2.3.1.5	cimOtherIdentifyingInfoInstanceNameForm	2.3.2