INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Diplomarbeit

Entwurf eines Managementobjektmodells für ATM–basierte Netze

Bearbeiter:	Maximilian Fischer
Aufgabensteller:	Prof. Dr. Heinz-Gerd Hegering
Betreuer:	Boris Gruschke Norbert Wienold

INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Diplomarbeit

Entwurf eines Managementobjektmodells für ATM–basierte Netze

Bearbeiter:	Maximilian Fischer
Aufgabensteller:	Prof. Dr. Heinz-Gerd Hegering
Betreuer:	Boris Gruschke Norbert Wienold
Abgabetermin:	15. August 1999

Hiermit versichere ich, daß ich die vorliegende Diplomarbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 15. August 1999

....

(Unterschrift des Kandidaten)

Zusammenfassung

Diese Arbeit beschreibt den Entwurf eines Objektmodells für das Management von ATM-basierten Rechnernetzen und Client/Server-Diensten, insbesondere der *LAN Emulation (LANE)* und *Classical IP and ARP over ATM (CLIP)*, denen große Bedeutung für die Integration von ATM-Netzen in bestehende Netzinfrastrukturen zukommt.

Grundlage für dieses Modell bilden die Konzepte, die die *Distributed Management Task Force* (DMTF) in ihrer Spezifikation eines *Common Information Model* (CIM) [CIM 99a] festlegt. CIM dient der einheitlichen und herstellerunabhängigen Darstellung von Managementinformation für heterogene Systeme. Dazu wird ein objektorientierter Ansatz benutzt. Für verschiedene Bereiche des Managements gibt CIM bereits Objektmodelle vor. Die darin erfaßten akstrakten Konzepte können durch Vererbung anwendungsspezifisch verfeinert werden. Auf Basis dieser Modelle können dann sowohl Managementapplikationen entwickelt werden, die auf einer sehr hohen Abstraktionsebene operieren und so die Heterogenität der Systeme verschatten, als auch stark spezialisierte Anwendungen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Innerhalb der letzten Jahre hat sich in einer Vielzahl von Unternehmen die Güte der Netzinfrastruktur zu einem entscheidenden Erfolgsfaktor entwickelt. Schnelle und zuverlässige Kommunikation in Form von Daten, Sprache und vielerorts auch Videokonferenzen sind mittlerweile eine Selbstverständlichkeit. Auf Seiten der Applikationen, die die Unternehmensnetze nutzen, haben Client/Server-basierte Lösungen mit ansprechenden graphischen Oberflächen zunehmend Terminal-basierte Großrechneranwendungen abgelöst. Daneben finden Internet- und Intranet-basierte Anwendungen immer mehr Zuspruch, nicht zuletzt aus Gründen der Kosteneffektivität durch den möglichen Einsatz verhältnismäßig einfacher Arbeitsplatzsysteme, bekannt unter dem Schlagwort "thin clients". Durch den Einsatz elektronischer Groupware-Lösungen erhoffen sich viele Unternehmen Produktivitätssteigerungen. All diesen Beispielen ist eines gemeinsam: sie alle tragen dazu bei, daß sich die Anforderungen an das Unternehmensnetz erhöhen. Zum einen liegt dies an einem gesteigerten Bedarf an Bandbreite, zum anderen auch an Anforderungen an die Dienstgüte, die bestimmte Anwendungen—beispielsweise Videoübertragungen—für einen reibungsfreien Betrieb stellen.

Damit konfrontiert stehen viele Netzwerkverantwortliche vor dem Problem, die Infrastruktur den gestiegenen Anforderungen anzupassen. Die Wahl der Übertragungstechnologie, die dabei zum Einsatz kommen soll, hängt wesentlich von den konkret eingesetzten oder zukünftig geplanten Anwendungen ab. Steht hier die Möglichkeit einer Integration verschiedenartiger Dienste wie Sprach-, breitbandiger Video- und Datenübertragungen über ein *gemeinsames* Netz und die Garantie von Dientgütevereinbarungen im Vordergrund, so bietet der verbindungsorientierte *Asynchronous Transfer Mode (ATM)* entscheidende Vorteile gegenüber verbindungslosen Verfahren wie *Fast* oder *Gigabit* Ethernet. Kommt es dagegen in erster Linie auf eine Erhöhung der verfügbaren Bandbreite an, so sind die zuletzt genannten Verfahren aus Kosten- und Komplexitätsgründen in vielen Fällen zu favorisieren.

Eine weitere in der Praxis vermehrt eingesetzte Netzkonfiguration für größere Unternehmensnetze stellt die Kombination von traditionellen LAN-Technologien wie Ethernet im Endsystembereich und ATM im Backbonebereich dar. Das Breitbandnetz im Backbone sorgt dabei dafür, daß eine leistungsfähige Anbindung der Endsysteme an zentralisierte Serversysteme gewährleistet ist. Daß auf ATM bis hin zum Endsystem bei dieser Konfiguration verzichtet wird, liegt zum einen an den hohen Kosten und zum anderen an der mangelnden Verfügbarkeit reiner ATM-Anwendungen. Dies impliziert die Forderung nach einer Interoperabilität von ATM-Netzen und den Protokollen der LAN-Welt. Die beiden Ansätze, die hier am häufigsten eingesetzt werden, sind die *LAN Emulation* (LANE) und *Classical IP and ARP over ATM* (CLIP). Beide Verfahren sind über verteilte Client- und Serverdienste realisiert und gestatten die Bildung *virtueller* Netze, also verschiedener logischer Gruppierungen von Endsystemen basierend auf ein- und derselben physischen Netztopologie. Sie stellen somit beträchtliche Managementaufgaben bei Konfiguration und Überwachung des laufenden Betriebs.

Die verschiedenen Hersteller von ATM-Komponenten bieten hierfür in der Regel proprietäre Lösungen, die der Forderung nach einem integrierten Management, also einer einheitlichen Managementsicht auf heterogene Ressourcen im Rahmen einer standardisierten Architektur, widerspricht. Eine solche integrative Sicht, ausgedrückt durch ein *einheitliches Informationsmodell*, bietet unbestreitbare Vorteile für das Management und schafft erst die Grundlage für eine Vielzahl von Managementaufgaben, beispielsweise für Verfahren zur Event-Korrelation.

1.2 Ziele der Arbeit

Hauptziel der Arbeit ist die Identifikation *generischer* Managementinformation, die ein herstellerunabhängiges Management von ATM-basierten Netzen und Diensten wie LANE und CLIP ermöglicht. Diese erfolgt primär nach einem Top-Down Vorgehen, es werden aber auch bereits bestehende Lösungen zur Bereitstellung von Managementinformation einbezogen (Bottom-Up). Als Informationsmodell soll das objektorientierte *Common Information Model* (CIM) verwendet werden, das bereits eine Reihe von Referenzmodellen für das Netz- und Systemmanagement vorgibt. Ziel ist es, diese Modelle ATM-spezifisch zu erweitern und dabei eine aussagekräftige Modellierung der physischen, virtuellen und logischen Sichten auf das ATM-Netz sowie der Abhängigkeiten zwischen diesen Sichten zu erreichen. Zur Population und Nutzung dieses Informationsmodells wird eine CORBA-basierte Managementarchitektur skizziert.

1.3 Grundlagen zu ATM

Für das Verständnis der Arbeit wird vorausgesetzt, daß der Leser über gute Kenntnisse zur Funktionsweise von ATM sowie der LANE- und CLIP-Dienste verfügt. In dieser Arbeit werden lediglich einige wesentliche Sachverhalte wiederholt. Aufgrund der reichen Auswahl an guter Literatur zu ATM-Grundlagen [Black 95, Black 98a, Hein 97, Kyas 96, Pry 94, Tan 96] und zum Internetworking mit ATM [Black 98b, BHK 95] soll hier eine erneute Darstellung dieser Themenbereiche unterbleiben.

1.4 Aufbau der Arbeit

Das anschließende Kapitel 2 bietet einen Überblick über die Konzepte des Common Information Model (CIM). Insbesondere werden die wesentlichen Grundlagen für das Verständnis der graphischen und textuellen Darstellung von Managementobjektmodellen in CIM vermittelt und die Hauptbestandteile der Spezifikation erläutert.

Kapitel 3 stellt verschiedene Management-Lösungen für ATM-Netze und Komponenten vor. Besondere Beachtung finden dabei die Spezifikationen zum SNMP-Management privater ATM-Netze von ATM-Forum und IETF, die bisher allerdings nur zum Teil Eingang in konkrete Produkte gefunden haben. Die angebotene Managementinformation stellt die Grundlage der Bottom-Up Analyse für die späteren eigenen Modelle dar. Als Beispiel für herstellerspezifisches ATM-Management werden einige SNMP-MIBs für ATM-Komponenten der Firma FORE Systems und deren Managementapplikation FORE View betrachtet.

Kapitel 4 beschreibt den Weg zu einer ATM-spezifischen Erweiterung der CIM Basismodelle.

Kapitel 5 schließt die Arbeit mit einem Ausblick auf offene Fragestellungen ab.

Kapitel 2

Überblick über das Common Information Model (CIM)

Ziel dieser Arbeit ist ein Management-Objektmodell für ATM-basierte Rechnernetze. Zur Beschreibung dieses Modells wird das *Common Information Model* (CIM) [CIM 99a] der *Distributed Management Task Force* (DMTF, bis Mai 1999 unter dem Namen *Desktop* Management Task Force bekannt, der das ursprüngliche Hauptaufgabengebiet dieser Organisation im Bereich des Managements von Desktop PCs widerspiegelt) verwendet und erweitert.

Das vorliegende Kapitel stellt zunächst kurz Ursprung und Zielsetzung von CIM vor, bevor die fundamentalen Konzepte dieser Technologie beschrieben werden.

2.1 Grundsätzliches zu CIM

CIM dient der Beschreibung von Managementobjekten (*Managed Objects*) und deren statischen Beziehungen untereinander. Damit stellt CIM in erster Linie den Teil einer Managementarchitektur dar, der als *Informationsmodell* bezeichnet wird. Hierbei kommt konsequent der objektorienterte Ansatz zum Einsatz, dessen Vorteile im Hinblick auf eine aussagekräftige Modellierung der realen Welt und dabei insbesondere bei der Formulierung von Objektbeziehungen gegenüber einem einfachen Datentypansatz wie im Internet-Management (SNMP, *Simple Network Management Protocol*) bereits aus anderen Managementstandards (z.B. OSI, ODP) hinreichend bekannt sind (vgl. [HAN 99]).

Seinen Ursprung hat CIM in den Bemühungen der *Web-Based Enterprise Management* (WBEM) Initiative um ein Management für Rechnernetze und Endsysteme, das auf Internet-Technologien basiert. Die erarbeiteten Vorschläge gingen Anfang 1997 an die DMTF über, die für die Weiterentwicklung eine eigene Unterorganistation, das *CIM Sub-Committee*, ins Leben rief. In diesem Organ sind so namhafte Unternehmen wie IBM/Tivoli, Cisco, Compaq, Dell, Hewlett-Packard, Intel, Sun Microsystems und die Microsoft Corporation vertreten, womit das Potential für eine breite Akzeptanz der Entwicklungen vorhanden sein dürfte.

Erstes veröffentlichtes Ergebnis war die CIM Spezifikation in der Version 1.0, die im April

1997 erschienen ist. Besonderer Wert wurde dabei darauf gelegt, daß CIM bestehende Managementkonzepte wie SNMP oder DMI (*Desktop Management Interface*) nicht etwa ersetzen, sondern vielmehr integrieren soll: „CIM [...] attempts to *unify* and extend the existing instrumentation and management standards“ [CIM 98d]. Um dieses Ziel zu erreichen, sind Verfahren zur Abbildung bereits vorhandener Managementinformation (z.B. aus SNMP MIBs) in das CIM vorgesehen.

Im Juni 1999 wurde die CIM Spezifikation Version 2.2 [CIM 99a] vorgestellt. Es handelt sich dabei lediglich um eine Weiterentwicklung und Berichtigung der Version 2.0 [CIM 98a, CIM 98b], die fast zwei Jahre lang aktuell war. Die ursprünglich für die Folgeversionen angekündigten Neuerungen wie die Nutzung der *Extensible Markup Language* (XML) zur Web-basierten Darstellung von Managementinformation und die Definition einer eigenen Abfragesprache (*Query Language*) haben jedoch noch keinen Eingang in die Spezifikation gefunden.

Neben der bereits angesprochenen CIM Spezifikation existieren zu CIM auch eine Reihe von vorgegebenen, recht allgemein und herstellerunabhängig gehaltenen, Objektmodellen für verschiedene Bereiche des Managements. Dieser Satz von Modellen wird als *CIM Schema* [CIM 99b] bezeichnet und liegt mittlerweile ebenfalls in Version 2.2 vor.

2.2 Struktur von CIM

Dieser Abschnitt gibt eine Gliederung der Hauptbestandteile von CIM an. Dazu werden nacheinander die bereits angesprochenen Bestandteile CIM Spezifikation und CIM Schema betrachtet.

2.2.1 Die CIM Spezifikation v2.2

CIM Meta Schema

Hauptbestandteil der CIM Spezifikation ist die Definition des sogenannten *Meta Schemas*, das eine Formalisierung der zur Verfügung stehenden Modellierungsregeln durch ein übergeordnetes Modell darstellt. Die Struktur dieses Meta-Modells ist in der graphischen Form eines Klassendiagramms angegeben, vgl. Abbildung 2.1. Als Notation findet hier und durchweg in CIM die *Unified Modeling Language* (UML) Verwendung [RJB 98]. Die wesentlichen graphischen Primitive für die Erstellung von Klassendiagrammen werden im folgenden nach und nach erläutert. Eine umfassende Darstellung der Möglichkeiten der UML und deren Diagrammtypen würde den Rahmen dieser Arbeit sprengen. Als gute Einführung sei auf [Oest 98] verwiesen.

Das Meta Schema beinhaltet zunächst klassische Elemente der objektorientierten Modellierung:

- Eine *Class* definiert die Attribute und Operationen für eine Menge von Objekten gleichen Typs (Instanzen der Klasse). In UML-Notation wird eine Klasse durch ein Rechteck dargestellt, das vertikal in bis zu drei Bereiche aufgeteilt ist (vgl. Abbildung 2.2). Zwingend gefordert ist hier die Angabe eines Klassennamens im ober-

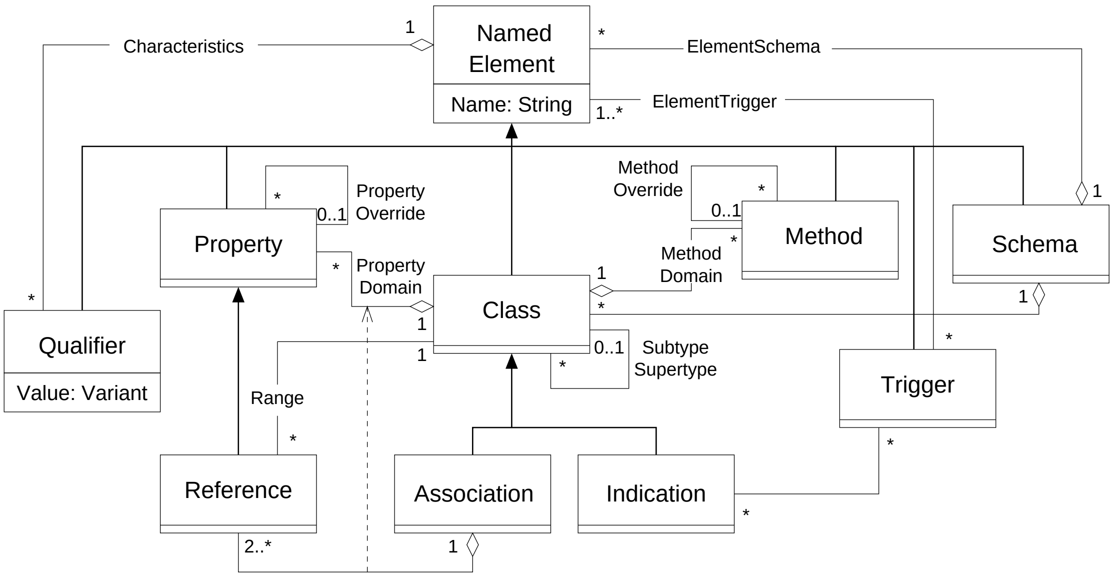

Abbildung 2.1: UML-Diagramm des CIM Meta Schema

sten Bereich, der die Klasse eindeutig identifiziert. Definiert die Klasse eigene Attribute (*Properties*) oder Methoden (*Methods*), so werden diese im (optionalen) zweiten bzw. dritten Bereich angeführt.
Ein sehr wichtiges Konzept in der Objektorientierung ist das der *Vererbung* (engl. *Inheritance*). Damit kann ausgedrückt werden, daß eine (Unter-)Klasse Attribute und Operationen von einer übergeordneten (Ober-)Klasse erbt. Dies ermöglicht eine effiziente Modellierung von hierarchischen Strukturen, insbesondere von Spezialisierungsbeziehungen. Das Meta Schema erlaubt nur eine *strikte Einfachvererbung*, d.h. Klassen besitzen jeweils maximal eine direkte Oberklasse und alle Attribute und Operationen der Oberklasse müssen auch in der Unterklasse definiert sein. Aus diesem Grund werden vererbte Attribute und Methoden im allgemeinen nicht explizit in der Notation der Unterklasse gezeigt, es sei denn, sie werden in der Unterklasse überschrieben (*override*).
In der UML werden Vererbungsbeziehungen durch einen Pfeil von der Unter- zur Oberklasse dargestellt. Aus Gründen der Übersichtlichkeit werden sie in den Diagrammen der offiziellen Dokumente zum CIM in blauer Farbe eingezeichnet. Dies stellt aber keine Maßgabe der UML dar, weshalb in dieser Arbeit darauf verzichtet wurde. In Abbildung 2.1 erbt beispielsweise die Klasse Class (auch: Property, Method, usw.) von der Oberklasse Named Element, die die Wurzel der Vererbungshierarchie des Meta Schemas markiert.

- Eine *Property* entspricht einem Attributwert, der eine bestimmte Eigenschaft einer Klasse beschreibt. In [CIM 99a] werden als zulässige Datentypen vorzeichenlose und vorzeichenbehaftete ganze Zahlen mit Wertebereichen zwischen 8 und 64 Bit (`uint`x bzw. `sint`x, $x = 8, 16, 32, 64$), IEEE Gleitkommazahlen (`real`x, $x = 32, 64$), Boolesche Werte (`boolean`), Unicode Zeichen bzw. Strings (`char16`, `string`), ein Datumsformat (`datetime`) und eine streng typisierte Referenz (`ref`) definiert. Beispielsweise ist die Property Name eine Eigenschaft von Objekten der

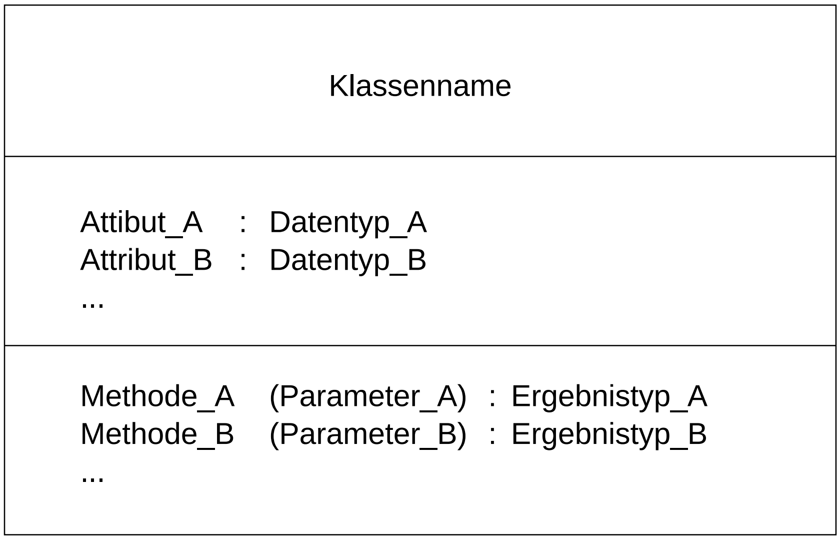

Abbildung 2.2: Graphische Notation einer Klasse in der UML

Klasse NamedElement mit dem Typ `String`.
Eine einzelne Klasse kann selbstverständlich über mehrere Properties verfügen. Im UML-Diagramm des Meta Schemas wird dies durch die (in offiziellen CIM-Dokumenten grün eingezeichnete) *Aggregationsbeziehung* zwischen Property und Class dargestellt. Aggregationsbeziehungen dienen der Modellierung von Teil-Ganzes-Beziehungen zwischen Klassen. Im vorliegenden Fall stellt Class das Ganze dar, notiert mittels der Rautenförmigen Pfeilspitze. Da die Aggregation einen Spezialfall der Assoziation darstellt, können auch hier Multiplizitäts- und Rollenangaben vorgenommen werden, wie dies weiter unten bei der Behandlung der *Association* beschrieben ist.

- Eine *Method* gibt die Signatur (Methodenname, Eingabeparameter und Typ des Rückgabewertes) einer Operation an, die einen Dienst auf Objekten der zugehörigen Klasse realisiert.

- Eine *Association* stellt allgemein eine Beziehung zwischen Klassen dar. Die bereits erwähnten Vererbungs- bzw. Aggregationsbeziehungen stellen Spezialfälle davon dar. Da Association eine Unterklasse von Class ist, sind Associations benannt und es können auch zu Associations Unterklassen per Vererbung gebildet werden, die wiederum Associations darstellen. Beispielsweise ist die Beziehung zwischen Reference und Association abgeleitet von der hierarchisch übergeordneten Beziehung zwischen Property und Class. Im Diagramm ist dies durch den gestrichelten Pfeil zwischen den genannten Beziehungen eingezeichnet. Im allgemeinen muß für gewöhnliche Associations (in den UML-Diagrammen von CIM rot gefärbt) und für Aggregationsbeziehungen auch eine *Multiplizität* (auch *Kardinalität* genannt) angegeben werden. Diese gibt die Anzahl der Objekte (konkrete Instanzen von Klassen) an, die auf beiden Seiten einer Association miteinander in Beziehung treten. Fehlt diese Angabe, so ist von einer 1:1-Beziehung auszugehen. Im UML-Diagramm erfolgt die Angabe der Multiplizität in numerischer Form an den Enden des jeweiligen Beziehungssymbols. Beispielsweise ist aus Abbildung 2.1 ersichtlich, daß jedes Objekt der Klasse Association *mindestens* zwei Objekte der Klasse Reference enthält. Der Name eines Objekts der Klasse Reference (vererbtes Attribut Name) gibt dabei die *Rolle* an, die das referenzierte Objekt in einer Beziehung einnimmt. In der UML werden Rollen an den Enden des Beziehungssymbols in textueller Form angegeben. Bei der UML-Notation des Meta Schemas aus Abbildung 2.1 wurden Rollennamen aus Platzgründen weggelassen. Daß dies nicht ganz unproblematisch ist zeigt der Umstand, daß z. B. in der Association SubtypeSupertye, die im Meta Schema die Vererbungsbeziehung modelliert, nicht unmittelbar klar ist, welcher Instanz der Klasse Class die Rolle der Oberklasse und welcher die der Unterklasse zukommt. Lediglich die oben gemachte Einschränkung im Hinblick auf das Verbot von Mehrfachvererbung löst durch Betrachtung der Kardinalitätsangaben dieses Problem auf.

Darüber hinaus beschreibt das Meta Schema auch Konzepte, die CIM-spezifische Bedeutung haben:

- Der im Zusammenhang von CIM häufig auftretende Begriff des *Schemas* (unter anderem ja bereits im Namen Meta Schema selbst) steht für eine logische Gruppierung zusammengehöriger Klassem. Innerhalb eines Schemas müssen Klassennamen eindeutig sein.
- Ein *Trigger* wird verwendet, um Zustandsänderungen bei Objekten erkennen zu können. Beispielsweise kann die Änderung eines Attributwertes (z.B. durch Aufruf einer entsprechenden Set-Methode von außen) einen Trigger auslösen.
- Eine *Indication* zeigt das Eintreten eines Ereignisses an. Normalerweise wird ein Indication-Objekt erzeugt, nachdem ein Trigger ausgelöst wurde.

  Zu beachten ist, daß weder Trigger noch Indications in (statischen) UML-Klassendiagrammen ausgedrückt werden können. Dies erfordert die Modellierung dynamischer Vorgänge in Form von Zustandsübergangsdiagrammen bzw. Sequenzdiagrammen (*Message Sequence Charts*).
- Ein *Qualifier* dient der näheren Charakterisierung von Named Elements. Jeder Qualifier hat einen Namen und einen Wert (eines vorgegebenen Datentyps). In [CIM 99a] werden vier verschieden Arten von Qualifiern definiert. Die (booleschen) *Meta*-Qualifiers `ASSOCIATION` und `INDICATION` sowie alle *Standard*-Qualifiers müssen von jeder CIM-konformen Implementierung korrekt interpretiert werden. Daneben gibt es noch *Optional*-Qualifiers, die ignoriert werden dürfen, und *User-defined*-Qualifiers. Letztere sollten nur in solchen Fällen benutzt werden, in denen das vorhandene Repertoire einen bestimmten Sachverhalt nicht ausdrücken kann.

  Beispiele für vorhandene Standard-Qualifiers sind der boolesche Qualifier `ABSTRACT` im Zusammenhang mit Klassen, um anzuzeigen, daß es sich um eine nicht instantiierbare Klasse handelt, oder der String-Qualifier `DESCRIPTION` der die Möglichkeit bietet, eine Freitext-Beschreibung zu Named Elements anzugeben.

Ein Großteil der Qualifier besitzt keine direkte Entsprechung im UML-Diagramm (Ausnahmen stellen z.B. der Qualifier `ABSTRACT`, der in der UML mit dem *Stereotyp* ≪abstract≫ beim Klassennamen ausgedrückt werden kann, oder der Qualifier `WEAK` bei Beziehungen (Erklärung dazu erfolgt im nächsten Abschnitt) dar, der in CIM-Diagrammen—als Erweiterung zur UML—mit dem Buchstaben „`w`“ beim entsprechenden Assoziationsende notiert wird). Sie haben jedoch entscheidende Bedeutung für die Semantik der beschriebenen Modelle und werden daher im *Managed Object Format* (MOF), der textuellen Form der Darstellung von Managementinformation in CIM, die im folgenden vorgestellt wird, sehr intensiv benutzt.

Managed Object Format (MOF)

Formal wird Managementinformation in CIM durch eine Notation in Textform definiert, die auf der *Interface Definition Language* (IDL) der Open Group basiert und als *Managed Object Format* (MOF) bezeichnet wird. Um den Austausch von Managementinformation zu begünstigen, kann die MOF-Syntax durch eine kontextfreie Grammatik angegeben werdem, was die Implementierung von MOF-Parsern ohne großen Aufwand gestattet. Die vollständige Syntax in BNF-Notation ist in [CIM 99a] angegeben, in diesem Abschnitt sollen einige Beispiele die wesentlichen Konstrukte erläutern.

Zur Definition einer Klasse dient das Schlüsselwort `class` (Groß- und Kleinschreibung sind im MOF ohne Bedeutung) gefolgt vom Namen der Klasse. Handelt es sich um eine abgeleitete Klasse, so steht nach einem Doppelpunkt der Name der Oberklasse. Danach folgen Attribut- und Methodendeklarationen in einer an die Sprache C++ angelehnten Syntax, Kommentare im Stil von C++ sind ebenfalls erlaubt, wie in Abbildung 2.3 gezeigt.

```
// Deklaration einer Klasse ´my_class´, die von ´my_parent_class´ erbt

class my_class : my_parent_class
{
  Datentyp_A Attribut_A = Defaultwert;
  Datentyp_B Attribut_B;

  Ergebnistyp_A Methode_A(Parametertyp_A Parametername_A);
  Ergebnistyp_B Methode_B(Typ_B1 Parametername_B1, Typ_B2 Parametername_B2);
};
```

Abbildung 2.3: Beispiel einer MOF–Klassenbeschreibung

Zu beachten ist, daß das MOF keine Vorwärtsreferenzen gestattet, d.h. alle in einer Klassendeklaration als Typen verwendeten Klassen müssen bereits definiert sein. Durch den Einsatz von Qualifiern können alle Named Elements (also Klassen, Attribute, Methoden und deren Parameter) näher bestimmt werden. In der MOF-Syntax werden Qualifier in eckigen Klammern dem jeweiligen Named Element vorangestellt. Mehrere Qualifier werden durch Kommata abgetrennt, soll einem Qualifier ein bestimmter Wert zugewiesen werden, so steht dieser unmittelbar folgend in runden Klammern. Beispielsweise wird ein Qualifier benötigt, um auszudrücken, daß es bei einer Klasse um eine Beziehung (Assoziation) handelt. Das Beispiel in Abbildung 2.4 stammt aus dem CIM Core Schema, auf das an späterer Stelle noch kurz eingegangen wird.

```
// ==================================================================
//    SystemComponent
// ==================================================================
        [Association, Aggregation, Description (
         "CIM_SystemComponent is a specialization of the CIM_Component "
         "association that establishes 'part of' relationships between "
         "a System and the Managed System Elements of which it is "
         "composed.") ]
class CIM_SystemComponent:CIM_Component
{
        [Override ("GroupComponent"), Aggregate,
         Description ("The parent System in the Association.") ]
    CIM_System REF GroupComponent;
        [Override ("PartComponent"), Description (
         "The child element that is a component of a System.") ]
    CIM_ManagedSystemElement REF PartComponent;
};
```

Abbildung 2.4: MOF-Beschreibung einer Assoziation aus dem Core Schema

Durch Voranstellen des Qualifiers `ASSOCIATION` wird hier erreicht, daß die Klasse CIM_SystemComponent als Beziehung zu interpretieren ist. Genauer gesagt handelt es sich um eine Aggregationsbeziehung—wobei eine Instanz der Klasse CIM_System mit dem Rollennamen ´GroupComponent´ das Ganze darstellt—was durch den Qualifier `AGGREGATION` angezeigt wird. Die `OVERRIDE` Qualifier sagen aus, daß die Deklaration der entsprechenden Attribute der Oberklasse CIM_Component in der Unterklasse außer Kraft gesetzt ist und neu definiert wird.

Große Bedeutung im Zusammenhang mit Assoziationen haben auch die Qualifier `MIN`, `MAX` und `WEAK`. `MIN` und `MAX` dienen der Festlegung der Multiplizität einer Beziehung. Dazu wird—selbstverständlich für beide Seiten der Beziehung separat—angegeben, wie viele Objektinstanzen *mindestens* bzw. *maximal* beteiligt sein müssen. So stellt die Assoziation CIM_HostedService (ebenfalls aus dem CIM Core Schema) eine 1:n-Beziehung dar, siehe Abbildung 2.5.

Damit wird in diesem Fall ausgedrückt, daß die CIM-Abstraktion eines „Service" auf *genau einem* „System" lokalisiert ist. Da für die Beziehung auf der Seite der Klasse CIM_Service keine Multiplizitätsangaben vorgenommen werden, finden hier die Vorbelegungen `0` für `MIN` und `NULL` für `MAX` Verwendung, womit auf dieser Seite beliebig viele Objektinstanzen stehen können. Ein „System" kann also beliebig viele „Services" beheimaten.

Der Qualifier `WEAK` bei CIM_Service bedeutet, daß diese Klasse von der anderen beteiligten Klasse (in diesem Fall CIM_System) *abhängig*, also relativ zu dieser *schwach* ist. Von besonderer Bedeutung ist dies für die Identifikation von Objektinstanzen. Dies erfolgt in CIM nicht wie bei vielen anderen objektorientierten Technologien über einen eindeutigen Objekt-Identifikator (z.B. eine IOR im Falle von CORBA), sondern mittels Schlüsselattributen, die in der MOF-Beschreibung durch Voranstellen des Qualifiers `KEY` definiert werden. Der Wert dieses Attributes beziehungsweise die Kombination aller vorhandenen Schlüsselattribute, falls mehrere definiert sind, muß eine Objektinstanz eindeutig

```
// ==================================================================
//    HostedService
// ==================================================================
        [Association, Description (
         "CIM_HostedService is an association between a Service and "
         "the System on which the functionality resides.  The "
         "cardinality of this association is 1-to-many.  A System may "
         "host many Services. Services are weak with respect to their "
         "hosting System. Heuristic:  A Service is hosted on the "
         "System where the LogicalDevices or SoftwareFeatures that "
         "implement the Service are located.  The model does not "
         "represent Services hosted across multiple systems.  This is "
         "modeled as an ApplicationSystem that acts as an aggregation "
         "point for Services, that are each located on a single "
         "host.") ]
class CIM_HostedService:CIM_Dependency
{
        [Override ("Antecedent"), Max (1), Min (1),
           Description ("The hosting System.") ]
    CIM_System REF Antecedent;
        [Override ("Dependent"), Weak,
         Description ("The Service hosted on the System.") ]
    CIM_Service REF Dependent;
};
```

Abbildung 2.5: MOF-Beschreibung einer Assoziation mit Multiplizitätsangaben

identifizieren. Schlüsselattribute entsprechen damit dem aus dem Datenbank-Bereich her bekannten Konzept des *Primärschlüssels* (vgl. z.B. [Date 95]). Ist eine Klasse durch eine entsprechende Assoziation einer anderen Klasse gegenüber als abhängig eingestuft, so bedeutet dies, daß deren Schlüsselattribute auch *notwendiger* Bestandteil der Identifikation von Objektinstanzen der abhängigen Klasse sind. Für die Klasse CIM_Service bedeutet dies beispielsweise, daß sie die Attribute SystemCreationClassName und SystemName erhält (siehe Abbildung 2.6) und diese zu den Schlüsselattributen zählen. In einer konkreten Instanz von CIM_Service werden diesen Attributen die Werte der Attribute CreationClassName und Name des zugeordneten Objekts der Klasse CIM_System zugewiesen. Im Detail sind diese Zusammenhänge in [CIM 98d] erläutert.

2.2.2 Das CIM Schema v2.2

CIM hebt sich von anderen Ansätzen zur Modellierung von Managementinformation vor allem auch dadurch ab, daß nicht nur Modellierungsregeln vorgegeben werden, sondern auch standardisierte Modelle für wichtige Anwendungsbereiche. Es handelt sich hierbei um das *CIM Schema* [CIM 99b], das seinerseits aus verschiedenen Schemata zusammengesetzt ist, die im folgenden kurz vorgestellt werden.

CIM Core Schema

Den Regeln des Meta Schemas folgend ist das *CIM Core Schema* spezifiziert, das generische Basisklassen für das Netz- und Systemmanagement in einer Hersteller- und Implementierungsunabhängigen Form vorgibt. Ein vereinfachtes UML-Diagramm des Core Schemas zeigt Abbildung 2.6.

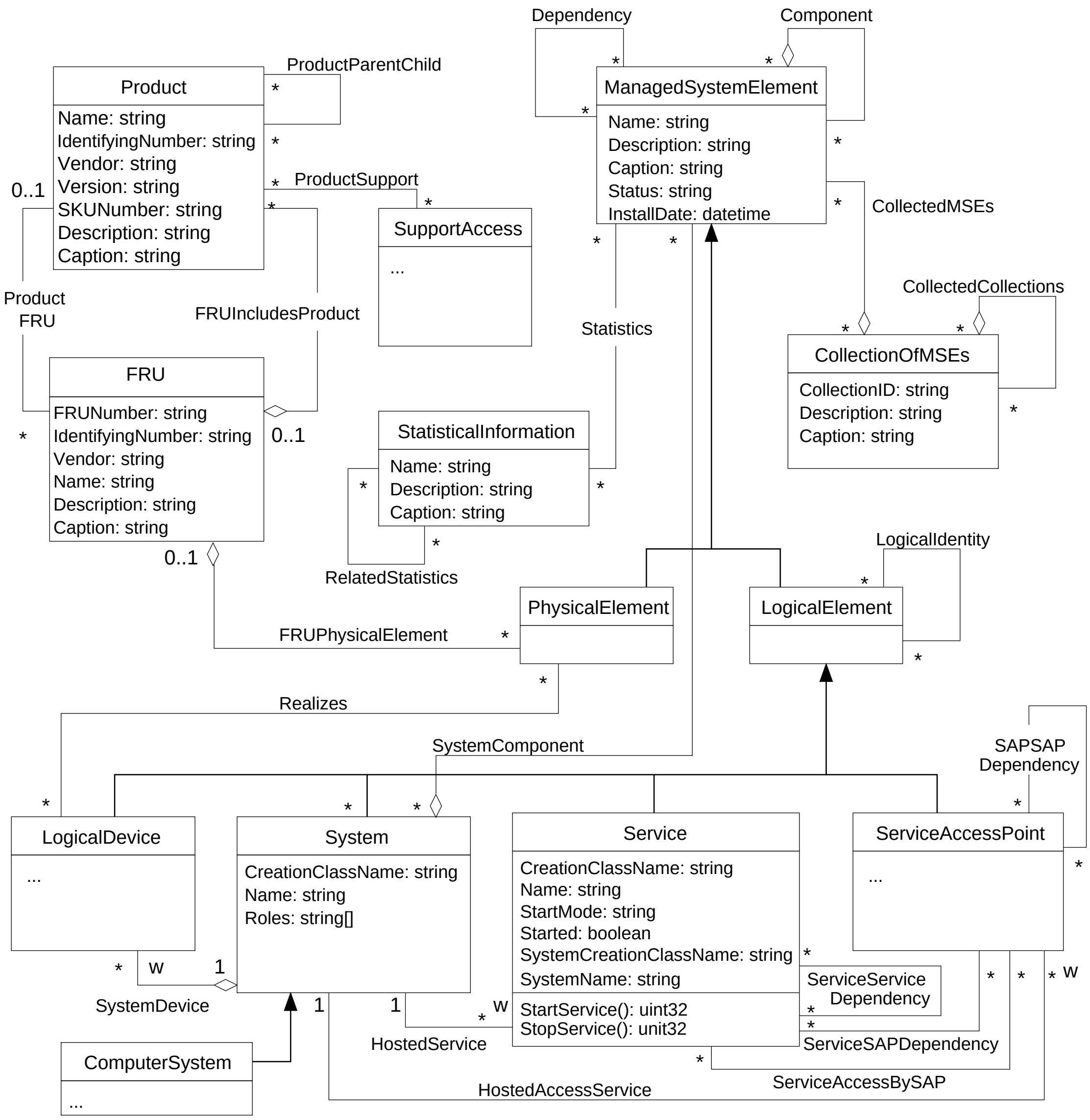

Abbildung 2.6: UML-Diagramm des CIM Core Schema (vereinfacht)

Zu den Konzepten des CIM Core Schemas existiert ein sehr ausführliches White Paper [CIM 98d], das über die WWW-Seite der DMTF (http://www.dmtf.org) erhältlich ist, womit sich eine genauere Vorstellung der einzelnen Klassen und Beziehungen an dieser Stelle erübrigt. Elemente des Core Schemas, die wesentliche Bedeutung für die Gestaltung der Modelle haben, die in Kapitel 4 beschrieben werden, werden dort im jeweiligen Zusammenhang aufgegriffen.

Die CIM Common Schemata

Darüber hinaus werden für besonders wichtige Anwendungsfälle des Managements—ebenfalls herstellerunabhängig—Grundlagen in Form der sogenannten *Common Schemata* gelegt. Im einzelnen handelt es sich hierbei um:

- *System Schema*
  (Management von Rechensystemen allgemein)
- *Device Schema*
  (Beschreibung systemnaher Hard- und Softwarekomponenten)
- *Physical Schema*
  (Modellierung der physischen Zusammensetzung von Hardware)
- *Application Schema*
  (Beschreibung des logischen Aufbaus von Software-Systemen)
- *Network Schema*
  (Modellierung von Rechnernetzen und Protokollen)

MOF-Dateien und UML-Diagramme im Format des Zeichenprogrammes *Visio* stehen auf der WWW-Seite der DMTF (http://www.dmtf.org) zum download bereit.

Im Gegensatz zum Core Schema sind aber—mit Ausnahme des Application Schemas—für diese Schemata keine einführenden Dokumente über die DMTF verfügbar. Als Interpretationshilfe ist man in erster Linie auf die Kommentare in den zugehörigen MOF-Dateien angewiesen.

Auf das Network Schema wird in Kapitel 4 noch genauer eingegangen, da es die Basis für die ATM-spezifische Erweiterung der CIM-Modelle bildet, die in dieser Arbeit vorgestellt wird. Für kommende Versionen von CIM ist außerdem ein *Database Schema* angekündigt.

CIM Extension Schemata

Die Common Schemata sind sehr allgemein gehalten und daher als Grundlage für eigene Weiterentwicklungen (im Sinne einer Verfeinerung) zu verstehen. Solche Erweiterungen werden als *Extension Schemata* bezeichnet. Beispielsweise hat die Firma Microsoft im Rahmen der WBEM Initiative für die 32-bit-Versionen des Betriebssystems *Windows* ein *Win32* Extension Schema zum System Schema publiziert [WBEM 98].

Kapitel 3

Bestehende Ansätze zum ATM-Management

Die Entwicklung von ATM wurde—da es sich um eine verhältnismäßig junge Technologie handelt—von Anfang an von Bestrebungen begleitet, ausreichende Schnittstellen für das Netzmanagement vorzusehen. Dieses Kapitel bietet einen Überblick über die entsprechenden Lösungen der beiden wesentlichen Standardisierungsgremien der ATM-Technologie, dem *ATM-Forum* und der *Internet Engineering Task Force* (IETF).

Neben den vorgestellten Standards existiert eine Vielzahl von herstellerspezifischen Lösungen. Exemplarisch werden hier einige MIBs und die Management-Plattform der Firma *FORE Systems* vorgestellt, einem der führenden Hersteller von ATM-Komponenten, dessen Produkte auch am Lehrstuhl für Rechnernetze der Ludwigs-Maximilians-Universität München zum Aufbau eines ATM-Testnetzes für das Rechnernetze-Praktikum [Sche 99] eingesetzt wurden.

3.1 Managementstandards des ATM-Forums

Maßgebendes Organ für die Standardisierung von ATM ist das *ATM-Forum*, ein Zusammenschluß wichtiger Hersteller von Netzkomponenten und Netzbetreibern, der seit seiner Gründung im Jahr 1991 auf mehrere hundert Mitglieder gewachsen ist, darunter so bekannte Konzerne wie Cisco, Bay Networks, FORE oder AT&T. Innerhalb dieser Organisation beschäftigt sich eine eigene Arbeitsgruppe, die *Network Management Sub-Working Group*, mit der Spezifizierung von Management-Lösungen für private und öffentliche ATM-Netze, wobei der Fokus der Arbeiten hier auf den öffentlichen Netzen liegt. Aber auch auf dem Gebiet der privaten ATM-Netze und der ATM-Dienste haben die Arbeitsgruppen des ATM-Forums Netzmanagementlösungen festgelegt, die im folgenden vorgestellt werden. Der nächste Abschnitt beschreibt das *Integrated Local Management Interface (ILMI)*, eine Managementschnittstelle, die Zugang zu Informationen über ATM-Schnittstellen auf Ebene der ATM-Übertragungstechnik bereitstellt. Im Anschluß daran werden verschiedene MIBs beschrieben, die für das Management von Diensten eingeführt wurden, die auf ATM basierenen. Besondere Aufmerksamkeit kommt dabei der *LAN Emulation (LANE)* zu, die eine große Bedeutung für die Integration von ATM-Netzen in konventionelle Netzumgebungen hat. Weitere Ansätze zu *Multiprotocol over ATM (MPOA)* und einer *Remote Monitoring (RMON)* MIB für ATM werden kurz vorgestellt.

3.1.1 Die ILMI MIB

Für das Management von Zugangsschnittstellen zu privaten ATM-Netzen, bezeichnet als *User Network Interfaces* (UNIs), hat das ATM-Forum eine Lösung in Form der *Integrated Local Management Interface (ILMI) Specification* [ILMI 96] vorgelegt. Ursprünglich sollte ILMI nur eine Übergangslösung darstellen und wurde daher zunächst als *Interim* Local Management Interface betitelt. Mittlerweile hat sich aber abgezeichnet, daß diese Schnittstelle bestand haben wird, so daß mit den Weiterentwicklungen zur aktuellen Version 4.0 auch eine Umbenennung erfolgte.

Als Kommunikationsprotokoll kommt bei ILMI das aus dem Internet-Management bekannte SNMP-Protokoll zum Einsatz. Dabei werden die SNMP-Protokolldateneinheiten aber nicht als UDP-Datagramme verschickt, sondern als AAL5-Frames an das ATM-Interface übergeben. Hierfür existiert ein reservierter virtueller Kanal (VPI=0, VCI=16) am UNI. ILMI erlaubt den Zugriff auf eine ILMI MIB, die ebenfalls in [ILMI 96] definiert ist, wobei die üblichen SNMP-Protokollinteraktionen *get-request*, *get-next-request*, *set-request* und *trap* (vgl. [HAN 99])) zur Verfügung stehen.

Die ILMI MIB ist in vier MIB-Module gegliedert:

- *Textual Conventions MIB*
  Hier werden einige grundlegende Objektdefinitionen getroffen, beispielsweise wird ein Datentyp `AtmAddress` vereinbart. Andere MIB-Module können dieses Basismodul importieren, womit für Konsistenz gesorgt ist.
- *Link Management MIB*
  Dient der Verwaltung physischer und virtueller Verbindungen an ATM-Interfaces. Beispielsweise können hier PVC-Verbindungen konfiguriert werden.
- *Address Registration MIB*
  Dieses MIB-Modul wird benötigt, damit ATM-Endsysteme das Netzwerk-Präfix ihrer ATM-Adresse via ILMI ermitteln können. Dieses Präfix wird im ATM-Switch vorkonfiguriert.
- *Service Registry MIB*
  Mit Hilfe der Service Registry MIB wird ein Mechanismus bereitgestellt, durch den angeschlossene Endsysteme ermitteln können, welche speziellen Dienste vom ATM-Netz für potentielle Dienstnutzer (in der Regel Endsysteme) an einem UNI angeboten werden. Zum Beispiel kann hier registriert sein, daß über das UNI ein ATMARP-Dienst erreichbar ist, wie in Abschnitt 3.2.2 angesprochen.

Außerdem muß jede Komponente, die die ILMI MIB implementiert, zusätzlich auch die *system* Gruppe der MIB-II [McRo 91] realisieren.

Die meiste Funktionalität in der ILMI MIB bietet das Link Management Module. Die angebotene Managementinformation umfaßt sowohl Angaben zu Schicht 1 (*physical layer*) und ATM-Schicht eines ATM-Interfaces, als auch Managementobjekte, die sich auf einzelne virtuelle Kanäle bzw. Pfade beziehen. Unter anderem sind hier auch Informationen über angeschlossene Nachbarsysteme zu finden, was für die automatische Ermittlung (*Autodiscovery*) der physischen Topologie eines ATM-Netzes in Managementapplikationen benötigt wird.

3.1.2 LANE Client und Server MIBs

Der Ansatz des ATM-Forums, um ATM und bestehende LANs auf Basis von Ethernet oder Token Ring zu integrieren ist die *LAN Emulation Over ATM (LANE)* [LANE 97]. Sie bietet Schicht-3-Protokollinstanzen eine virtuelle MAC-Schnittstelle zum ATM-Netz, so daß diese Protokolle (wie beispielsweise IP) unverändert funktionieren, ohne ATM-Charakteristika—wie z.B. die Verbindungsorientiertheit—berücksichtigen zu müssen. Realisiert wird diese Emulation durch einen Client/Server-Ansatz. Das emulierte MAC-Interface stellt dabei den *LANE Client (LEC)* dar, der sich zur Erbringung dieser Funktionalität auf drei Server-Dienste abstützt:

- *LAN Emulation Configuration Server (LECS)*
  Dieser Server ordnet jedem LEC einen LES zu und bestimmt damit die Zugehörigkeit zu einem *Emulated LAN (ELAN)*.
- *LAN Emulation Server (LES)*
  Dieser Server ist für eine Gruppe von zusammengehörigen LE Clients zuständig, die ein ELAN bilden. Hauptaufgabe des LES ist die Adreßauflösung zwischen MAC-Adressen und ATM-Adressen über eine Adreßtabelle.
- *Broadcast and Unknown Server (BUS)*
  Dieser Server dient der Bereitstellung der Broadcast-Funktionalität, die für die herkömmlichen *shared media* LAN Technologien charakteristisch ist. Dazu unterhält er eine Punkt-zu-Mehrpunkt Verbindung zu allen LE Clients innerhalb eines ELANs, über die Broadcast-Nachrichten weitergeleitet werden.

Konfiguration und Überwachung dieser Client- und Server-Dienste stellt erhebliche Anforderungen an das Netzmanagement. Das ATM-Forum hat zu diesem Zweck zwei Spezifikationen [LECM 97, LESM 96] für das SNMP-Management von LANE Implementierungen veröffentlicht. [LECM 97] befaßt sich mit dem Management von LE Clients und definiert dazu eine *LAN Emulation Client MIB*, deren Implementierung das ATM-Forum für alle Systeme, auf denen LE Clients realisiert sind, empfiehlt. Das zweite Dokument ([LESM 96]) führt analog dazu drei MIB Module für die verschiedenen Server-Dienste ein.

LAN Emulation Client MIB (LEC MIB)

Diese MIB enthält eine Reihe von Managementobjekt-Gruppen, die die Konfiguration, Verwaltung und Überwachung von LE Clients gestatten. Besonders wichtig sind dabei folgende Gruppen, die als SNMP-Tabellen organisiert sind, wobei jede Tabellenzeile für einen bestimmten Client steht:

- *Configuration Group*
  Durch Modifikation dieser Tabelle können LE Clients erzeugt oder gelöscht werden. Neue LE Clients können hier auch mit ihren *Initial State Parameters* [LANE 97] konfiguriert werden, wie beispielsweise der Angabe der ATM-Adresse des zuständigen LECS oder der Art der gewünschten Emulation (Ethernet oder Token Ring).

- *Status Group*
  Liefert (read–only) Status-Informationen zu den LE Clients. Ein Großteil dieser Information entspricht den Initial State Parameters aus [LANE 97].

- *Statistics Group*
  Hier werden Statistiken geführt, die zum Beispiel die Anzahl von Adreßauflösungswünschen oder die Anzahl von gescheiterten Verbindungsaufbauversuchen zu anderen Clients angeben.

- *Server Connections Group*
  Diese Tabelle identifiziert die verschiedenen virtuellen Verbindungen, die LE Clients zu ihren Server-Diensten unterhalten.

- *LE_ARP Cache Group*
  Über diese MIB–Gruppe kann der Cache verwaltet werden, der bereits bearbeitete Adreßauflösungswünsche zwischen MAC und ATM-Adressen (LE_ARP) für eine spätere Wiederverwendung speichert.

Emulated LAN MIB (ELAN MIB)

Bevor LE Clients aktiv werden können, müssen sie sich in einem ELAN registrieren. Dabei können Richtlinien (*policies*) eingesetzt werden, die vorgeben, welche Clients an einem bestimmten ELAN teilnehmen dürfen (beispielsweise auf Basis der ATM-Adresse). Mit Hilfe der ELAN MIB können solche ELANs samt Richtlinien angelegt und konfiguriert werden. Im allgemeinen nutzt diese Informationen der LECS-Dienst, der allerdings bei einer festen Vorkonfiguration aller LE Clients nicht zwingend erforderlich ist. Drei weitere Gruppen in der ELAN MIB mit Bezug zum LECS sind daher optional:

- *LECS Configuration Group*
  An dieser Stelle können die Werte konfiguriert werden, die der LECS auf Konfigurationsanfragen von LE Clients liefert (vgl.[LANE 97]). Außerdem lassen sich bestehende virtuelle Verbindungen zu LE Clients identifizieren.

- *LECS Statistics Group*
  Bietet Zähler mit Angaben zu erfolgreichen und erfolglosen Konfigurationsanfragen beim LECS. Im letzteren Fall werden verschiedene Fehlerfälle unterschieden und getrennt gezählt.

- *LECS Fault Management Group*
  In der LECS Fault Management Group kann der Netzwerkverantwortliche ein *logging* von Fehlern, die bei Anfragen an den LECS auftreten können, veranlassen. Ist diese Funktion aktiviert, so werden Fehler mit Zeitstempel, Fehlerursache und ATM-Adresse des Clients, der den Fehler verursachte, in einer Tabelle vermerkt.

LAN Emulation Server MIB (LES MIB)

Mit Hilfe der LES MIB können LE Server erzeugt, konfiguriert und überwacht werden. Unter anderem lassen sich alle LE Clients (und virtuelle Verbindungen zu diesen) identifizieren, die einen bestimmten LES benutzen, um so die Topologie eines ELANs zu ermitteln, und die Adreßauflösungstabelle (zwischen MAC- und ATM-Adressen) verwalten. Analog zur ELAN MIB bietet auch die LES MIB *Statistics* und *Fault Management* Gruppen.

Broadcast and Unknown Server MIB (BUS MIB)

Die BUS-MIB dient der Verwaltung von BUS-Diensten. In ähnlicher Weise wie bei der LES MIB können auch hier Topologieinformationen (Auflistung aller LE Clients, die ein bestimmter BUS bedient und zugehörige virtuelle Verbindungen), BUS Statistiken und Fehlerprotokolle abgerufen werden.

3.1.3 Die MPOA MIB

Mit der *Multiprotocol over ATM (MPOA)* Spezifikation [MPOA 99] hat das ATM-Forum ein gegenüber der LAN Emulation weiterentwickeltes Verfahren zur Übertragung bestehender (*legacy*) Protokolle über ATM vorgestellt. Es handelt sich dabei ebenfalls um einen Client/Server-basierten Ansatz, dessen Komponenten die *MPOA Clients* (MPC) und *MPOA Server* (MPS) darstellen. Vorteilhaft im Vergleich zur LANE ist dabei, daß MPOA Clients aus verschiedenen Subnetzen—beispielsweise IP-Subnetzen, falls IP als Vermittlungsschicht-Protokoll dient (wobei MPOA aber grundsätzlich nicht auf ein bestimmtes Protokoll ausgelegt ist)—unter Umgehung von Routern *direkt* miteinander kommunizieren können, falls das zugrundeliegende physische ATM-Netz dies gestattet. Dadurch ist ein entscheidender Performanz-Gewinn zu erwarten. Für das SNMP-Management der MPCs und MPS hat das ATM-Forum eine eigene *MPOA MIB* [MPOA 98] entwickelt, die das Pendant zu den entsprechenden LANE MIBs darstellt.

Da MPOA ein recht neuer Ansatz ist, ist ein konkreter Einsatz in der Praxis noch sehr selten. Dies liegt unter anderem auch an der mangelnden Verfügbarkeit von Implementierungen. Insbesondere sehen die Hersteller oft keine Möglichkeit vor, bereits vorhandene ältere ATM-Hardware mit Firmware-Updates für einen Einatz in MPOA-Umgebungen aufzurüsten. Aus diesem Grund wird MPOA in dieser Arbeit nicht weiter betrachtet. Für eine genaue Darstellung der Funktionsweise sei auf [Black 98b] verwiesen.

3.1.4 Die ATM RMON MIB

Eine weitere neue Spezifikation zum ATM-Management mittels SNMP stellt die *ATM RMON MIB* [RMON 97] des ATM-Forums dar. Bei der RMON (*Remote Monitoring*) MIB, beschrieben im RFC 1271 [Wald 91], handelt es sich um eine Erweiterung des SNMP-Managements dahingehend, daß die bereitgestellten Managementobjekte „übergeordnete“ Objekte darstellen. Damit gemeint ist Managementinformation, deren Beschaffung vergleichsweise hohe Anforderungen an den Management-Agenten stellt, der die MIB implementiert. Dazu zählt beispielsweise das autonome Ermitteln von statistischen Daten durch den Agenten. Dies steht zwar in gewissem Sinne im Widerspruch zu einem Ziel des Internet-Managements, nämlich nur sehr einfache Agenten zu benötigen, stellt aber einen ersten Ansatz in Richtung eines *Funktionsmodells* dar. Eine genauere Beschreibung der RMON MIB, deren Einordnung in das SNMP-Management sowie eine Bewertung findet sich in [HAN 99].

Die RMON MIB aus RFC 1271 ist für das Management im Ethernet-Bereich ausgelegt. Mit der ATM RMON MIB stellt das ATM-Forum eine ATM-spezifische Erweiterung vor, die unter anderem folgende Managementinformation anbietet:

- *ATM Statistics Group*
  Diese Gruppe bietet statistische Zähler—beispielsweise zur Anzahl von Verbindungsaufbauwünschen—an, jeweils bezogen auf eine ausgewählte Schnittstelle oder eine Schnittstellen-Gruppe.

- *ATM Host Group*
  Dient der Ermittlung des Netzverkehrs, den bestimmte Hosts verursachen.

- *ATM Matrix Group*
  Zählt den Netzverkehr zwischen allen überwachten Hosts und liefert damit eine Verkehrsmatrix.

- *ATM MatrixTopN Tables*
  Ermittelt und liefert die Hosts, die am meisten zum Netzverkehr beitragen.

Damit bietet die ATM RMON MIB im Vergleich zur ILMI MIB und zur AToM MIB, die im folgenden Abschnitt zu Management-Lösungen der IETF behandelt wird, eine deutlich höhere Funktionalität und erlaubt es dem Netzverwalter, an aussagekräftige Managementdaten zu gelangen, ohne seinerseits größeren Netzverkehr zu deren Beschaffung zu verursachen, wie dies bei ständiger Abfrage (*polling*) von Zählerständen der Fall wäre. Allerdings ist die MIB in den meisten ATM-Produkten bislang noch nicht implementiert. In absehbarer Zeit dürfte sich dies jedoch ändern.

3.2 Arbeiten der IETF zum ATM-Management

Neben dem ATM-Forum wirkt auch die IETF an der Standardisierung von ATM mit, insbesondere mit Blickrichtung auf den Einsatz im Internet-Umfeld. Mittlerweile wurde eine ganze Reihe von *Request For Comments (RFCs)* zu ATM publiziert, von denen die wichtigsten in Bezug auf das Netzmanagement hier kurz betrachtet werden.

3.2.1 RFC 2515 — AToM MIB

RFC 2515 [TeEd 99] *„Definitions of Managed Objects for ATM Management“* stellt den Hauptbeitrag der IETF für das SNMP-basierte Management von ATM Schnittstellen und Einzelkomponenten, aber auch von ATM-basierten Diensten und Netzen dar. Das Dokument definiert eine MIB, die an Position {`mib-2 37`} im ISO Internet-Registrierungsbaum angeordnet ist. Je nach Quelle wird diese MIB zumeist als *AToM MIB*, manchmal aber auch nur als *ATM MIB* bezeichnet. RFC 2515 ersetzt eine ältere Spezifikation aus RFC 1695 [AhTe 94] in einer aufwärtskompatiblen Weise, indem die MIB lediglich ergänzt, die alte Struktur aber beibehalten wurde. Einige nicht mehr benötigte Managementobjekte wurden jedoch als *obsolete* gekennzeichnet.

Diesem RFC wird mit RFC 2514 [NST 99] ein Begleitdokument zur Seite gestellt, das im wesentlichen *Textual Conventions* für ATM definiert, also Bezeichner auf Basis der SMI Standardtypen definiert, deren Semantik klarer auf ATM abgestimmt ist. Beispielsweise können so ATM-Adressen als `AtmAddr` anstatt als `OCTET STRING` angegeben werden. Man beachte, daß hier bereits ein anderer Bezeichner gewählt wird als in der oben beschriebenen ILMI MIB.

Managementobjekte für das Interface-Management

Die AToM MIB stellt Managementinformation für ATM-Interfaces bereit, die die Angaben der *interface* Gruppe der MIB-II ATM-spezifisch erweitern. Dazu wird eine SNMP-Tabelle `atmInterfaceConfTable` definiert, in der für jedes vorhandene ATM-Interface (gekennzeichnet durch einen individuellen Index) eine Tabellenzeile vorgesehen ist. Die Einträge umfassen beispielweise Informationen zur maximal möglichen und zur derzeitigen Anzahl von virtuellen Kanal-/Pfadverbindungen an einem Interface und die VCI/VPI-Identifikatoren des virtuellen Kanals, der an diesem Interface für das ILMI-Protokoll verwendet wird.

Darüber hinaus spezifiziert RFC 2515 auch die Interpretation der MIB-II *interface* und *system* Gruppen im Zusammenhang mit ATM.

Managementobjekte für das Verbindungs-Management

Für das Management von virtuellen Pfaden und Kanälen sind in der AToM MIB vier Gruppen von Managementobjekten vorgesehen:

- *ATM Interface Virtual Path Link (VPL) Group*
- *ATM Interface Virtual Channel Link (VCL) Group*
- *ATM Interface Virtual Path (VP) Cross Connect Group*
- *ATM Interface Virtual Channel (VC) Cross Connect Group*

Die ersten beiden Gruppen dienen dem Erzeugen, Modifizieren oder Löschen von (bidirektionalen) virtuellen Pfad- bzw. Kanalverbindungen (VPLs/VCLs) zum Aufbau von PVCs. Dazu wird jeder VPL (VCL) durch einen Eintrag in einer SNMP-Tabelle repräsentiert. Die entsprechende Tabellenzeile enthält dazu Informationen zu den verwendeten Identifikatoren (VPI bzw. VPI/VCI), zum Status der Verbindung und für beide Übertragungsrichtungen eine Referenz auf die vereinbarten Verkehrsparameter. Außerdem ist ein *CrossConnectIdentifier* eingetragen, der gegebenenfalls als Index in die unten beschriebenen Tabellen der *Cross Connect* Gruppen dient. Das Anlegen/Löschen von Verbindungen erfolgt durch das Erzeugen/Löschen von Tabelleneinträgen.

Stellt ein VPL/VCL nicht das Ende einer virtuellen Verbindung dar, sondern lediglich ein Teilstück entlang einer solchen Verbindung, so ist es erforderlich, die entsprechende Umsetzung der Pfad- und Kanalidentifikatoren sowie von Eingangs- zu Ausgangsinterface festzulegen. Dazu dienen die beiden *Cross Connect* Gruppen.

Da SNMP nur einfache Datentypen und Tabellen bietet, aber keine direkte Modellierung von Beziehungen zwischen Managementobjekten, gestaltet sich das Verbindungsmanagement mit den genannten SNMP-Tabellen der AToM MIB recht umständlich und unübersichtlich. Außerdem deckt sich die angebotene Managementinformation der AToM MIB beim Verbindungsmanagement teilweise mit der der ILMI MIB. Ein weiteres Problem ergibt sich für das Ende-zu-Ende Management von virtuellen Verbindungen: Da die MIB nur lokale Information liefert, müssen alle an der virtuellen Verbindung beteiligten Komponenten einzeln angesprochen werden.

3.2.2 RFC 2320 — IPOA MIB

Zur Übertragung von IP-Verkehr über ATM-Netze schlägt die IETF in RFC 2225 [LaHa 98] das Verfahren *„Classical IP and ARP over ATM“*, kurz *CLIP*, vor. Derzeit besitzt RFC 2225 den Status eines *Proposed Standard* und ersetzt den bisherigen Vorschlag zu CLIP von RFC 1577 [Laub 94] aus dem Jahr 1994, der die Grundlage der meisten bestehenden CLIP-Implementierungen bildet.

CLIP ermöglicht die Bildung von *logischen* IP-Subnetzen (LIS) auf einem physischen ATM-Netz. Zur Realisierung kommt dabei ein Client/Server-Ansatz zum Einsatz. Zentrale Komponente ist der sogenannte *ATMARP*-Server, der die Aufgabe der Adreßauflösung zwischen den ATM-Adressen und IP-Adressen innerhalb eines LIS übernimmt. Dazu wird eine Umsetzungstabelle verwaltet, die zu einer IP-Adresse aus dem Adreßbereich des entsprechenden LIS die zugehörige ATM-Adresse des angesprochenen LIS Clients bereitstellt.

RFC 2225 fordert im Gegensatz zur (älteren) CLIP-Spezifikation aus RFC 1577 ausdrücklich Managementvorkehrungen für Implementierungen von CLIP [LaHa 98]:

> „Clients built to this specification MUST implement and provide a Management Information Base (MIB) as defined in “Definitions of Managed Objects for Classical IP and ARP Over ATM Using SMIv2”.“

Die angesprochene MIB (genauer: SNMP-MIB) wird in einem eigenständigen Dokument, nämlich RFC 2320 [GLWK 98], definiert und dort als *„Classical ARP and IP over ATM (IPOA) MIB“* bezeichnet. Sie gliedert sich in drei Bereiche, die im folgenden kurz vorgestellt werden.

Basic Support MIB Definitions

Die Definitionen in der *Basic Support MIB* müssen sowohl von CLIP-Clients als auch von ATMARP-Servern unterstützt werden. Sie enthalten unter anderem Managementobjekte (genauer SNMP-Tabellen), die die folgenden Informationen anbieten und in erster Linie für das Konfigurationsmanagement wichtig sind:

- Zugehörigkeit des Systems zu Logical IP Subnetzen (LIS)
- Zuordnung von Subnetzadressen zu ATM-CLIP-Interfaces
- Benutzte virtuelle Kanäle (PVCs oder SVCs) für die Kommunikation zwischen LIS Clients

Client Supported MIB Definitions

Die *Client Supported MIB* stellt den Client-spezifischen Teil der IPOA MIB dar. Sie definiert eine SNMP-Tabelle (`ipoaArpClientTable`), die Informationen zu allen LIS Clients enthält, die auf dem zu managenden System vorhanden sind. Durch Manipulation der Tabelle können neue Clients erzeugt oder nicht mehr benötigte Clients gelöscht werden.

Server Supported MIB Definitions

Die *Server Supported MIB* muß nur von Systemen implementiert werden, auf denen mindestens ein ATMARP-Server-Dienst zur Verfügung steht. Sie enthält die Definition einer SNMP-Tabelle zur Festlegung und Überwachung des ATMARP-Servers für ein LIS. Ebenfalls ist es möglich, einem LIS mehrere ATMARP-Server zuzuordnen oder die Tabelle zu modifizieren und damit dynamisch LIS (definiert über den zugehörigen ATMARP-Server) zu erzeugen oder zu löschen. Für das Fehlermanagement sind außerdem die SNMP-Traps `ipoaDuplicateIpAddress`, `ipoaLisCreate` und `ipoaLisDelete` definiert, die asynchrone Ereignismeldungen bei der (versehentlichen) doppelten Vergabe von IP-Adressen bzw. beim Erzeugen/Löschen von CLIP Subnetzen darstellen.

3.3 Proprietäres ATM-Management am Beispiel FORE

Nicht zuletzt aufgrund der Tatsache, daß die Standardisierung des ATM-Managements in vielen Fällen noch nicht abgeschlossen ist, liefern viele Hersteller von Netzkomponenten eigene MIBs zu ihren Produkten. Grundlage bildet dabei in der Regel das Internet-Management auf Basis von SNMP. Dieser Abschnitt stellt verschiedene MIBs des Herstellers FORE Systems kurz vor und betrachtet abschließend die Managementapplikation *FORE View*, die diese Managementinformation nutzt.

3.3.1 FORE-spezifische MIBs

Die Firma FORE Systems hat für ihre ATM-Interfacekarten und ATM-Switches eine Reihe von proprietären MIBs entwickelt. Ihre Einordnung in den Internet-Registrierungsbaum zeigt Abbildung 3.1.

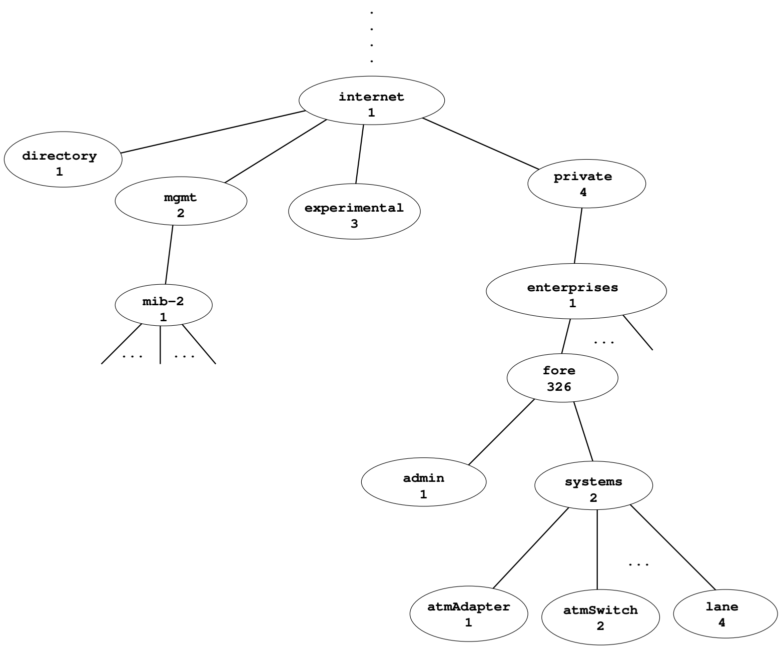

Abbildung 3.1: Einordnung der FORE-MIBs in den Internet-Registrierungsbaum

Der `atmAdapter` Teilbaum enthält MIB-Gruppen zu Produktinformationen (Typ, Firmware-Version, etc.), für das Schichtenmanagement der Schichten PHY, ATM und AAL, geschalteten SVC/PVC-Verbindungen, Status- und Statistikinformation (z.B. Fehlerzähler) sowie zur Konfiguration und Verwaltung von CLIP-Schnittstellen und ATMARP-Servern.

Im `atmSwitch` Teilbaum findet sich Managementinformation zur Hardware-Zusammensetzung und Ausstattung von ATM-Switches (Schnittstellenmodule, vorhandene Ports, CPU, Lüftung, etc.), zur Verwaltung der Switching-Tabelle, Topologie-Information (benachbarte Switches/Endsysteme) und zu den implementierten Signalisierungsprotokollen (neben Q.2391 auch das proprietäre SPANS [Fore 96a]). Daneben wird eine Vielzahl von SNMP-Traps (asynchrone Ereignismeldungen vom Agenten an einen Manager) definiert, die über Ereignisse wie Topologie-Änderungen (z.B. Anschluß eines Endsystems an einen Switch-Port) oder Fehlfunktionen (z.B. Ausfall des Lüftersystems) informieren.

Der Teilbaum `lane` dient der Administration der LAN Emulation. Er stellt MIB-Gruppen zum Erzeugen, Konfigurieren und Überwachen von LE Clients (LECs) und LE Servern (LECS, LES bzw. BUS) zur Verfügung. Im Vergleich zu den standardisierten LANE Client und Servers MIBs (vgl. Abschnitt 3.1.2) ist diese FORE-eigene Lösung recht einfach gehalten, natürlich bedingt durch die Tatsache, daß eine genau definierte Implementierung zugrunde liegt.

In der für das Internet-Management typischen Weise stellt die Managementinformation in den beschriebenen MIBs die Herstellersicht auf die eigenen Komponenten dar (gemäß dem Bottom-up-Vorgehen zur Gewinnung von Managementinformation). Die bereitgestellte Information deckt sich zudem in weiten Teilen mit der Information, die auch die bereits vorgestellten standardisierten MIBs anbieten, die die Komponenten der Firma FORE zum Teil ebenfalls bereits implementieren. Dies führt zu redundanter Darstellung von Managementinformation.

3.3.2 Die Managementapplikation *FOREView*

Als graphisches Hilfsmittel für das Management ihrer ATM-Komponenten bietet die Firma FORE Systems die Managementapplikation *FOREView* an. Diese kann sowohl einzeln (*stand-alone*) oder integriert in die Managementplattform *HP Open View* betrieben werden.

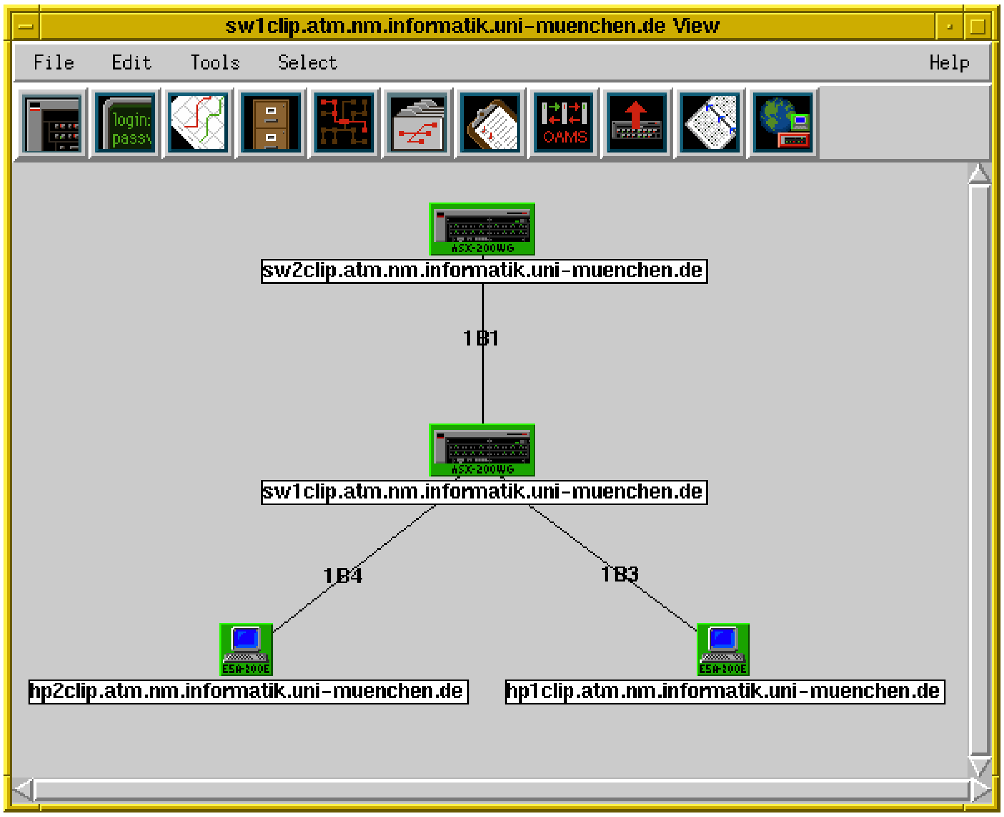

Abbildung 3.2: Darstellung einer ATM-Netztopologie in *FOREView* (`fvmap`)

Alle wesentlichen Funktionen (bzw. Funktionsgruppen) von *FOREView* sind über die Buttons der Bedienleiste des Programmes `fvmap` zu erreichen, dessen graphische Oberfläche in Abbildung 3.2 dargestellt ist.

FOREView ermöglicht die automatische Ermittlung (*auto-discovery*) und graphische Darstellung der vorhandenen physischen ATM-Topologie. Dabei werden zunächst nur ATM-Switches und deren physischen Verbindungen zueinander dargestellt. Wählt man ein Switch-Symbol per Doppel-Klick aus, so wird dieser Switch als aktuelle zentrale Komponente in einer neuen Sub-Map dargestellt und es werden auch die dort angeschlossen Endsysteme gezeigt. Ein einfacher Klick auf ein Symbol selektiert dieses.

Im einzelnen bietet *FOREView* die folgenden Funktionen (von links nach rechts im Toolbar):

- *Front Panel*
  Diese Funktion stellt die Bedienfront des selektierten ATM-Switches graphisch am Bildschirm dar. So können beispielsweise die Betriebsstatus-Anzeigen der einzelnen Switch-Ports von einer entfernten Managementstation aus inspiziert werden.

- *Telnet*
  Öffnet eine Telnet-Sitzung zu dem selektierten Switch. Auf diese Weise kann die auf den FORE-Switches verfügbare (Terminal-basierte) Managementsoftware *AMI (ATM Management Interface)* [Fore 96a] von einer Managementstation aus benutzt werden.

- *Graph*
  Graphische Darstellung des anfallenden Datenverkehrs für Switches, Endsysteme oder einzelne Verbindungen.

- *Log*
  Erfassung des anfallenden Datenverkehrs für Switches, Endsysteme oder einzelne Verbindungen in Form von Log-Dateien.

- *Trace*
  Startet das *Channel/Path Trace Tool*. Damit kann der Verlauf von virtuellen Verbindungen (über mehrere Switches hinweg) verfolgt werden. Die Darstellung erfolgt textuell anhand der beteiligten Port-Nummern und VPI/VCI-Identifikatoren

- *Configure*
  Startet das *Channel/Path Tool*. Dieses dient zur Konfiguration von virtuellen Verbindungen.

- *Inventory*
  Ermittelt selbsttätig Produktinformationen zur vorhandenen FORE-ATM-Hardware (Modell, Firmware-Version, etc.)

- *OAM Cell Monitor*
  Dient der Auswertung von OAM (*Operation and Maintenance*) Zellflüssen zur Analyse von Verbindungsproblemen. Wird nicht von allen FORE-Switches unterstützt.

- *Upgrade Switch*
  Mit dieser Funktion kann die Firmware von ATM-Switches beim Erscheinen einer weiterentwickelten Version aktualisiert werden.

- *Call Record*
  Sammelt Abrechnungs-Information. Wird nicht von allen FORE-Switches unterstützt.

- *VLAN Manager*
  Startet den *FOREView VLAN Manager*. Dieses Tool gestattet eine graphische Verwaltung von VLANs (auf herstellereigenen LAN-Switches) und ELANs per *drag-and-drop*. Abbildung 3.3 zeigt die dafür vorgesehene Oberfläche. Das abgebildete Beispiel stammt aus dem gemeinsamen Rechnernetze-Praktikum von LMU und TUM. Die Praktikumsanleitungen und [Sche 99] enthalten weitere Informationen zum VLAN-Manager und dessen Bedienung.

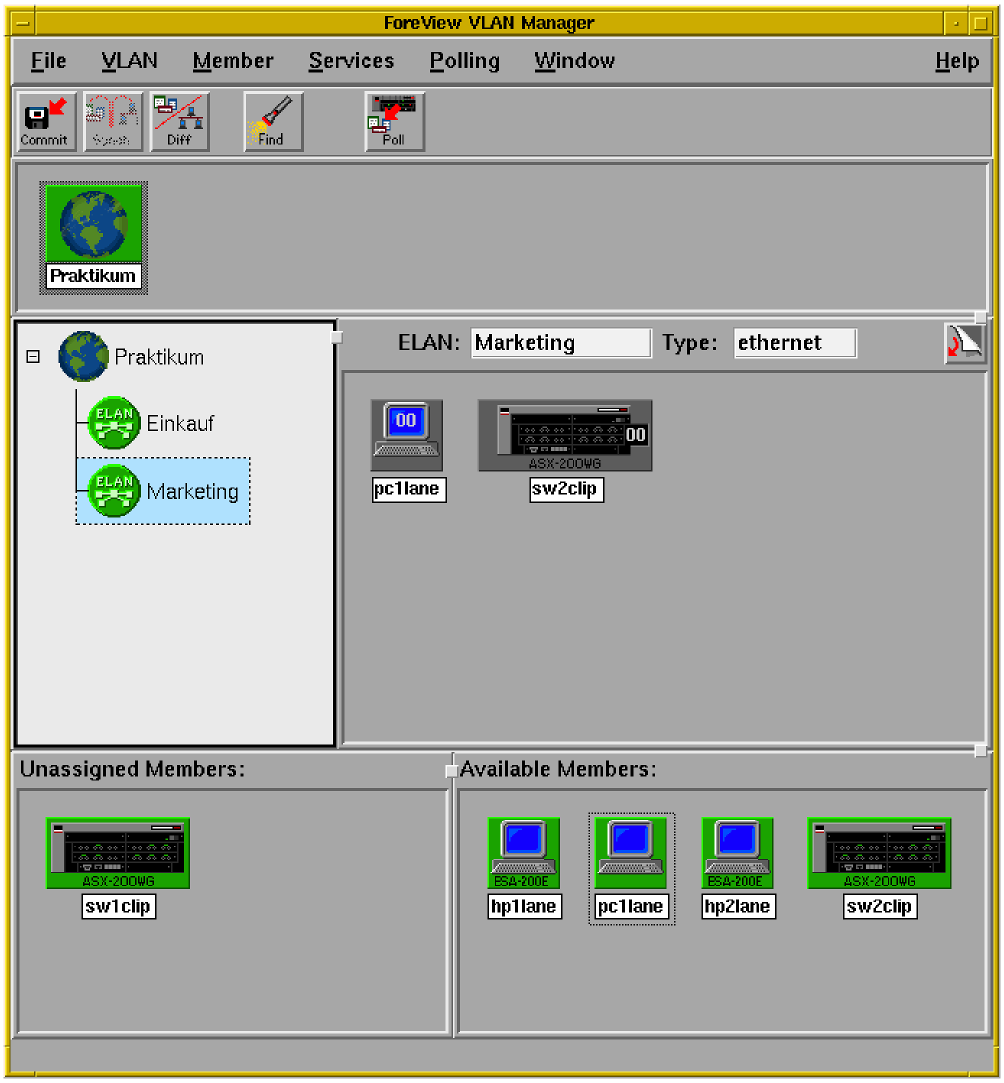

Abbildung 3.3: Verwaltung von ELANs mit dem *FOREView VLAN Manager* (`fvlan`)

Abschließend betrachtet stellt *FOREView* eine gelungene Lösung zum Management von FORE-Produkten dar, die viele Funktionen unter einer ansprechenden gemeinsamen Oberfläche vereint. Dies kann jedoch nicht darüber hinwegtäuschen, daß es sich um eine rein herstellerspezifische Insel-Lösung handelt, die für die Belange des integrierten Managements nicht geeignet ist. Einen weiteren Kritikpunkt stellt dar, daß mit Hilfe von *FOREView* zwar VLANs und ELANs (graphisch) verwaltet werden können, nicht jedoch beispielweise CLIP-Subnetze (LISs). Dies kann nur über Kommandozeilen-Befehle geschehen. Für die Bewältigung vergleichbarer Managementaufgaben, hier z.B. die Zuordnung von Endsystemen zu einem virtuellen Netz, sind also sehr unterschiedliche Benutzer-Schnittstellen vorgegeben.

Kapitel 4

Konzeption einer ATM-Erweiterung zum CIM

Bei der Konzeption eines Managementobjektmodells stellt sich zunächst die Frage: „was genau soll modelliert werden?“ Im gegebenen Fall ist das vorgegebene Ziel ein *generisches*, d.h. herstellerunabhängiges, Modell für das Management von ATM-basierten Client/Server-Diensten wie LANE und CLIP. Dieses Modell soll es ermöglichen, virtuelle Netztopologien, die mit Hilfe dieser Dienste realisiert werden, zu verwalten und eine Abbildung hin zu den beteiligten logischen Komponenten (ein ELAN besteht beispielsweise aus den Komponenten LECS, LES und BUS sowie mehrerer LECs) und den physischen Systemen, die diese realisieren, zu ermöglichen. Für das Konfigurations- und Fehlermanagement ist es von besonderer Bedeutung, die Beziehungen zwischen den beteiligten Komponenten zu erfassen. Eine Fragestellung könnte beispielsweise lauten: „laufen alle Server-Dienste, die ein bestimmtes ELAN realisieren?“

Um diese Forderungen zu erfüllen, müssen geeignete Managementobjekte und deren Beziehungen zueinander identifiziert werden. Dies kann mit Hilfe von Techniken der objektorientierten Analyse geschehen (vgl. z.B.[Oest 98]). Unverzichtbare Quellen bei diesem Arbeitsschritt sind natürlich auch die Spezifikationen zu den einzelnen Diensten LANE [LANE 97] und CLIP [LaHa 98]. Gute Anhaltspunkte finden sich auch in den zugehörigen Management-Spezifikationen, die bereits in Kapitel 3 betrachtet wurden.

Bei der Entwicklung ist außerdem zu beachten, daß der Modellierung ein bereits vorhandes Objektmodell, in diesem Fall das CIM Network Schema, zugrunde gelegt werden soll. Auf diese Weise soll die entstehende Managementlösung der Forderung nach einem einheitlichen Informationsmodell, das das CIM vorgibt, genügen. Aus diesem Grund ergeben sich folgende weitere wichtige Fragestellungen: „welche benötigten Konzepte sind bereits in ausreichender Form in den vorgegebenen Modellen enthalten?“ und „welche Stellen des vorgegebenen Modells eignen sich als Ansatzpunkte für eine Spezialisierung?“. Zur Beantwortung dieser Fragestellungen ist natürlich die Kenntnis des vorgegebenen Modells erforderlich. Der folgende Abschnitt widmet sich daher dem CIM Network Schema. Im Anschluß wird dieses dann geeignet erweitert.

4.1 Betrachtung des CIM Network Schema

Die DMTF gliedert das CIM Network Schema aus Gründen der Übersichtlichkeit in folgende drei Teilmodelle:

- *Network–BGP Sub-model*
(Allgemeine Grundlagen zur Modellierung von Netzen und Protokollen und insbesondere zur Darstellung und zum Austausch von Routing-Information. Spezielle Beachtung findet die Darstellung von Managementinformation zum *Border Gateway Protocol* (BGP), z.B. die Erfassung von *Routing policies*, vgl.[Tan 96])
- *Multiprotocol-Bridge Sub-model*
(Modellierung von Bridges bzw. Switches und zugehöriger Dienste, wie z.B. zur Steuerung von *Spanning Tree*-Mechanismen)
- *VLAN Sub-model*
(Modellierung der Bildung *virtueller LANs (VLANs)* mit VLAN-fähigen LAN-Switches)

UML-Diagramme zu diesen Teilmodellen (und dem CIM Core Schema) sind in Anhang A abgebildet, die zugehörige MOF-Beschreibung (hier wird nicht nach Teilmodellen unterschieden) in Anhang B. Alle Teilmodelle basieren auf Basisklassen des CIM Core Schemas und einigen Spezialisierungen davon, die im Network–BGP Teilmodell definiert sind. Diese werden im Anschluß beschrieben. Der Großteil der verbleibenden Teilmodelle ist für dieses Arbeit von untergeordneter Bedeutung und es wird daher nur noch kurz auf die Modellierung von VLANs eingegangen. Da alle in diesem Abschnitt erwähnten Klassen (und Beziehungen) aus dem CIM Schema stammen, wird im folgenden auf die Schemakennung „CIM_“ als Präfix des Klassennamens verzichtet.

4.1.1 Basisklassen für das CIM Network Schema

Den Ausgangspunkt für die Vererbungshierarchie im CIM Network Schema bilden folgende abstrakte Klassen aus dem Core Schema:

- System dient dazu, eine Menge von Managementobjekten, die zusammen eine funktionale Einheit bilden, als solche zu identifizieren. Im wesentlichen handelt es sich also um eine Aggregation von Komponenten, ausgedrückt durch die Aggregationsbeziehung SystemComponent. Insbesondere kann es sich bei solch einer Komponente auch um die Abstraktion von (physisch vorhandener oder emulierter) Hardware, ein LogicalDevice handeln. Dieser Fall wird durch die spezialisierte Beziehung SystemDevice modelliert. Konkrete Ausprägungen der abstrakten System-Klasse stellen beispielsweise die Unterklassen ComputerSystem und AdminDomain dar. Ein ComputerSystem stellt die Abstraktion eines Rechnersystems dar. Dabei kann es sich sowohl um einen Computer im geläufigen Sinn, als auch um ein dediziertes System wie z.B. einen Router handeln. Eine AdminDomain faßt alle Ressourcen zusammen, die unter einer gemeinsamen Verwaltung stehen. Alle Unterklassen von System tragen die Schlüsselattribute CreationClassName und Name. Für jede

Instanz einer von System abgeleiteten Klasse muß also die Kombination der Attribute CreationClassName und Name eindeutig sein. CreationClassName steht dabei für den Typ der Instanz, also eine der Unterklassen von System. Auf diese Weise können verschiedende Arten von Systemen bei gleichem Wert des Attributes Name voneinander unterschieden werden.

- Service stellt die managementrelevante Abstraktion von Funktionalität dar. Die HostedService-Beziehung gibt an, auf welchem System die Funktionalität lokalisiert ist. Services können nur im Kontext eines Systems existieren, deshalb trägt diese Beziehung den `WEAK` Qualifier auf Seiten von Service. Dies hat außerdem zur Folge, daß die Schlüsselattribute des zugehörigen Systems auch für die Identifikation des Services notwendig sind und von diesem übernommen werden. Funktionalität kann von Software (SoftwareFeature) oder Hardware LogicalDevice erbracht werden, modelliert über die Beziehungen SoftwareFeatureServiceImplementation bzw. DeviceServiceImplementation. Das Attribute StartMode legt fest, ob ein Service beim Systemstart automatisch (`Automatic`) oder nur auf Anforderung (`Manual`) gestartet wird. `StartMode` zeigt an, ob ein Service derzeit läuft (`TRUE`) oder nicht (`FALSE`). Über die generischen Methoden startService() und stopService() können Services gestartet bzw. angehalten werden, falls der konkrete Dienst dies zuläßt.

- ServiceAccessPoint modelliert die Zugangsschnittstelle zu einem Dienst. Steht ein Objekt der Klasse ServiceAccessPoint (SAP) zur Verfügung, so können andere Objekte auf den zugeordneten Service zugreifen. Diese Zuordnung erfolgt über die ServiceAccessBySAP-Beziehung. Wie Services sind auch Service Access Points eindeutig einem System zugeordnet (HostedAccessPoint-Beziehung).

- LogicalDevice stellt die managementrelevante Abstraktion einer (möglicherweise emulierten) Hardwarekomponente dar, beispielsweise einer Schnittstellenkarte oder eines Druckers.

- CollectionOfMSEs stellt ein sehr allgemeines Konzept zur Gruppierung von Managementobjekten dar. Unterklassen von CollectionOfMSEs sind logische Gruppen zusammengehöriger Objekte, die auf diese Weise als Einheit behandelt werden können.

Das Network–BGP Teilmodell enthält einige Grundklassen, die diese sehr allgemeinen Konzepte spezialisieren und sie so für die Modellierung von Rechnernetzen und Kommunikationsprotokollen anpassen. Die für den Rest der Arbeit wichtigsten davon sind:

- NetworkService bildet die Abstraktion von Diensten, die ein Rechnernetz zur Verfügung stellt. Aus diesem Grund handelt es dabei um eine Verfeinerung der Klasse Service. Beispielsweise stellt jegliche Form von Transportfunktionalität (z.B. Switching oder Routing) oder auch Hilfsdienste zur Erbringung dieser Funktionalität (beispielsweise der Austauch von Routing-Information) einen Network Service dar. Die Attribute StartupConditions und StartupParameters bieten eine generische Möglichkeit zur Definition von Vorbedingungen und zur Übergabe von Parametern beim Systemstart, die konkrete Implementierungen der (von Service vererbten) startService()-Methode in spezialisierten (abgeleiteten) Klassen auswerten können.

- ProtocolEndpoint stellt die Zugangsschnittstelle zu einem Network Service dar. In der MOF-Beschreibung heißt es dazu: „A communication point from which data may be sent or received“. Es handelt sich hier also um eine generische Kommunikationsschnittstelle. Solche Kommunikationsschnittstellen können von Software (Protokollschnittstellen) oder Hardware (z.B. Interfacekarten in Endsystemen oder Router-Ports) zur Verfügung gestellt werden. Unterklassen von ProtocolEndpoint sind beispielsweise IPProtocolEndpoint und LANEndpoint. IPProtocolEndpoint stellt den Zugangspunkt zu einer IP-Protokollinstanz dar, charakterisiert über deren IP-Adresse und Netzmaske (Attribute Address bzw. AddressMask). LANEndpoint modelliert eine MAC-Schnittstelle. Um auszudrücken, daß eine (oder mehrere) Protokollinstanz(en) der Schicht $(N+1)$ die Dienste einer (oder mehrerer) Protokollinstanz(en) der Schicht N nutzt, kann zwischen Protocol Endpoints die BindsTo-Beziehung angelegt werden. Dabei wird $(N+1)$-Instanzen der Rollenname Dependent, N-Instanzen der Rollenname Antecedent zugewiesen. Eine Spezialisierung dieser Beziehung stellt BindsToLANEndpoint dar.

- LogicalNetwork ist eine Verfeinerung der Klasse CollectionOfMSEs und modelliert die Gruppierung von Protocol Endpoints zu Subnetzen. Ausgedrückt wird diese Gruppierung durch eine Aggregationsbeziehung (z.B. InSegment). Beispiele für konkrete Logical Networks sind LANSegment und IPSubnet. Letzteres modelliert ein IP-Subnetz. Der Datenaustausch zwischen IP Protocol Endpoints kann darin ohne die Dienste von Routern erfolgen. LANSegment wird in der Klassenbeschreibung als „collection of LAN Endpoints of a particular type that are able to intercommunicate directly without the assistance of bridging or routing services.“ beschrieben. Diese Definition trifft beispielsweise auf ein *shared–medium* Ethernet-Segment zu, nicht jedoch auf ein beliebiges LAN, da hier mehrere Segmente über Kopplungselemente verbunden sein können.

- ActiveConnection ist eine Beziehungsklasse, die eine Verbindung zwischen Protocol Endpoints modelliert. Beispielsweise könnte so eine TCP-Verbindung modelliert werden. In der MOF-Beschreibung steht zudem die Forderung „This association defines a connection that is currently carrying traffic“. Diese Einschränkung erscheint jedoch etwas fragwürdig in Bezug auf (kurzzeitig) inaktive, aber dennoch fortbestehende Verbindungen. Das Attribut TrafficType dient der Klassifizierung des Datenverkehrs (mögliche Werte sind z.B. `Unknown`, `Unicast`, `Broadcast` oder `Other`, in letzterem Fall beinhaltet das Attribut `OtherTrafficDescription` eine nähere Beschreibung). Das boolesche Attribut isUnidirectional gibt an, ob die Verbindung uni- oder bidirektional ist.

Den Rest des CIM Network Schema bilden spezialisierte Klassen für weitverbreitete Kommunikationsprotokolle, Netzwerktechnologien und -dienste. Exemplarisch soll hier noch kurz auf das VLAN Teilmodell eingegangen werden.

4.1.2 Das VLAN Teilmodell

Das VLAN Teilmodell, dessen UML-Diagramm in Abbildung 4.1 dargestellt ist, orientiert sich sehr stark an den Vorgaben aus der Spezifikation für VLANs nach IEEE802.1Q

[IEEE 98]. Die VLAN-Funktionalität *eines Switches* repräsentiert die Klasse VLANService. Herstellerspezifische Lösungen werden durch Ableitung von dieser Klasse gebildet, vorgegeben ist lediglich ein 802dot1QVLANService. Für jedes VLAN, das ein Switch realisiert, ist ein Objekt der Klasse VLAN zu erzeugen. Dabei kann ein VLAN aber durchaus von mehreren VLANServices realisiert sein, beispielsweise, wenn ein Switch sowohl VLANs nach IEEE 802.1Q als auch nach einem proprietären Verfahren beherrscht. Die Assoziationen InboundVLAN und OutboundVLAN dienen der Festlegung von Filterregeln, die das VLAN definieren. Dies kann sowohl Port-basiert als auch nach MAC-Adressen oder Protokollfeldern erfolgen. Auffällig ist, daß keinerlei Methoden zur Manipulation der VLAN-Zugehörigkeit angeboten werden.

4.2 Erweiterung des CIM Network Schemas um ATM-Konzepte

Nachdem nun die wichtigsten Bestandteile des CIM Network Schemas erläutert sind, kann das Modell ATM-spezifisch erweitert werden. Die folgenden Abschnitte greifen jeweils einen bestimmten Teilaspekt auf und erläutern den aus dessen Analyse heraus entstandenen Modellteil. Aus Gründen der Übersichtlichkeit werden in den meisten nachfolgenden UML-Diagrammen nicht die kompletten Vererbungshierarchien des CIM Schemas abgebildet, sondern nur von einer Hierarchieebene an, die im jeweiligen Fall als sinnvoll erscheint. Eine gute Orientierungsmöglichkeit zur Einordnung dieser Modellteile in das CIM Gesamtmodell bieten die reduzierten UML-Diagramme zum (allgemeinen) Network Schema in Abbildung 4.2 (eine vergrößerte Darstellung davon befindet sich im Anhang A) und VLAN-Teilmodell in Abbildung 4.3. Diese Diagramme wurden mit dem CASE-Tool *Software through Pictures* (vgl. Kapitel 5) erstellt und bilden auch die Basis für die spätere Realisierung. Man beachte, daß das Hinzufügen oder Entfernen von Assoziationen laut CIM Spezifikation ([CIM 99a], Seite 5) die Interfaces der beteiligten Klassen nicht verändert. Dies rechtfertigt die weitestgehende Beschränkung auf Beziehungen, die auch tatsächlich direkt oder in abgeleiteter Form Verwendung finden.

4.2.1 Physische Sicht auf ein ATM-Netz

Physisch gesehen besteht ein ATM-Netz aus ATM-Switches und Endsystemen mit ATM-Interface sowie den Verbindungen zwischen diesen. In CIM stellt jede Form von Switch oder Router ein (dediziertes) ComputerSystem dar (vgl. CIM Core MOF, Anhang B.1) (die Switching-*Funktionalität* ist dagegen ein ForwardingService). Diese Definition kann auf ATM-Switches ausgeweitet werden. Es wäre möglich, eine Klasse ATMSwitch von ComputerSystem abzuleiten, die Identifikation als ATM-Switch kann aber auch über die Attribute Dedicated und Roles (vererbt von der Oberklasse System) erfolgen. Endsysteme werden ebenfalls durch (Unterklassen von) ComputerSystem modelliert. Die Modellierung eines ATM-Adapters erfordert die Verfeinerung der CIM-Klasse NetworkAdapter (vgl. CIM Device MOF, Anhang B.3), wie sie Abbildung 4.4 zeigt. Zu beachten ist, daß es sich bei ATMAdapter nicht um eine standardisierte Klasse aus den CIM Schemata handelt, sondern bereits um eine Erweiterung des Modells. Daher wird ein neuer Schema-Identifikator eingeführt. Diese und alle folgenden Erweiterungen des Modells er-

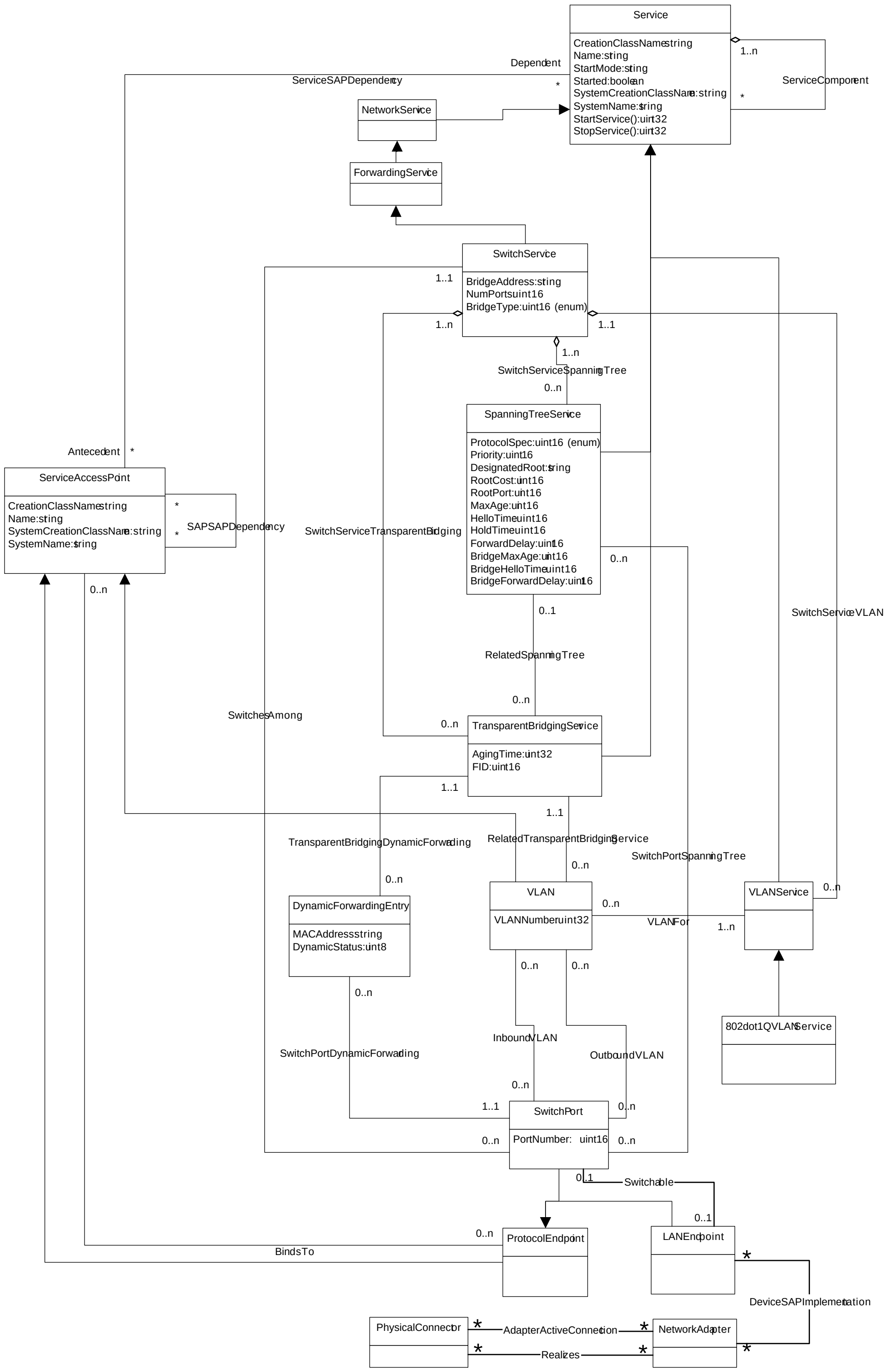

Abbildung 4.1: UML-Diagramm des VLAN-Modells im CIM Network Schema

halten das Präfix „CAE_". Dieses Kürzel ist freilich beliebig gewählt und steht für CIM ATM Extension. Im Folgenden wird auf die Kennungen „CIM_" bzw. „CAE_" verzichtet, solange keine Verwechslungsgefahr besteht.

Die Attribute der Klasse ATMAdapter orientieren sich an den bereits vorgegebenen Spezialisierungen EthernetAdapter und TokenRingAdapter sowie am Entwurf einer MIF (*Management Information Format* der DMTF) Beschreibung für ATM-Adapter [IBM 98]:

- NetworkAddresses gibt die ATM-Adressen an, die dem Adapter zugewiesen sind.
- AddressType gibt das Format der ATM-Adressen an. Mögliche Werte sind `Other`, `Private`, `NSAP` und `E.164` (vgl. [Kyas 96]), denen in dieser Reihenfolge Integerwerte von 0 bis 2 zugeordnet werden. Dieser Mechanismus entspricht den `Values`/`ValueMap`-Qualifiern, die in CIM für die Realisierung von Enumerationstypen verwendet werden.
- MaxVPIBits und MaxVCIBits gibt die Anzahl der Bits an, die an diesem Interface für VPI- bzw. VCI-Identifikatoren vorgesehen sind (variiert zwischen NNIs und UNIs).
- MaxVPCs und MaxVCCs gibt an, wieviele virtuelle Pfad- bzw. Kanalverbindungen maximal an diesem Interface konfiguriert werden können.
- ConfiguredVPCs und ConfiguredVCCs gibt deren tatsächliche Anzahl an.
- ILMI_VPI und ILMI_VCI gibt den virtuellen Kanal an, über den die ILMI-Schnittstelle an diesem Interface erreicht werden kann.
- HECErrors, HeaderErrors und InternalErrors zählen verschiedene Fehlerfälle wie Zellen mit korrumpiertem Header, illegalen VPI/VCI Identifikatoren oder Fehler bei der Weitergabe an höhere Schichten.

Ein ATMAdapter kann über die SystemDevice-Aggregationsbeziehung einem ComputerSystem zugeordnet werden. Wie Abbildung 4.1 zeigt, ist einem NetworkAdapter über die AdapterActiveConnection-Beziehung ein PhysicalConnector (Abstraktion einer Buchse oder eines Steckers) zuzuordnen. Diese wiederum können über einen PhysicalLink (Abstraktion eines Kabels) verbunden sein. Somit kann die physische Topologie erfaßt werden. Nähere Details zu den angeführten Klassen sind der MOF-Definition des CIM Physical Schema (vgl. Anhang B.5) zu entnehmen.

4.2.2 Modellierung ATM-spezifischer Dienstzugangspunkte

Dienste stellen ihre Funktionalität an Dienstzugangspunkten zur Verfügung. Dies trifft auch auf ATM-Komponenten zu, deren Schnittstellen durch ATM-Adressen charakterisiert sind. In Analogie zu LANEndpoint und IPProtocolEndpoint wird daher ein ATMEndpoint durch Ableitung von der Oberklasse ProtocolEndpoint definiert, wie Abbildung 4.5 zeigt (auf die Klassen LECEndpoint und IPOAClientEndpoint wird erst später im Zusammenhang mit LANE und CLIP eingegangen). Neben der ATM-Adresse (ATMAddress) und deren Typ (AddressType) erbt ATMEndpoint weitere Attribute von ProtocolEndpoint, unter anderem die Attribute Type und Name. Für Type ist in der aktuellen CIM Spezifikation v2.2 bereits ein Wert „`ATM`" vorgesehen (übrigens ist dies neben AdminDomain die einzige Stelle in CIM, an der ATM explizit erwähnt wird). Name gibt einen symbolischen Namen für das Interface an.

Die Zuordnung von ATMEndpoint-Objekten zu Systemen geschieht über die HostedAccessPoint-Beziehung. Zusätzlich wird die BindsToATMEndpoint-Beziehung eingeführt, die von BindsTo abgeleitet ist und aussagt, daß der ATMEndpoint von höheren Protokollschichten genutzt wird.

Logische Verbindungen (virtuelle Kanäle) zwischen ATMEndpoint-Objekten können über die ActiveConnection-Beziehung modelliert werden. Eine mögliche Form der Modellierung ist die Ableitung einer Beziehungsklasse ActiveATMConnection mit Attributen wie `sourceVPI`, `sourceVCI`, `targetVPI`, `targetVCI` und `QoSParameters`. Dies erscheint jedoch nur für PVC-Verbindungen wirklich sinnvoll. SVCs bereiten wegen ihrer inhärenten Dynamik Konsistenzprobleme.

4.3 Modellierung virtueller Netze

Virtuelle Netze gestatten die Bildung von logischen Subnetzen, ohne daß deren Topologie zwingend von der zugrundeliegenden physischen Struktur vorgegeben ist. Über einem physischen Netz können viele verschiedene solcher logischen Gruppierungen realisiert werden. Das passende Konzept im CIM Network Schema, das diese Umstände widerspiegelt, ist die Klasse Logical Network, die eine Gruppierung von Protocol Endpoints realisiert. CIM gibt als spezielle Logical Networks die Klassen IPSubnet, IPXNetwork und LANSegment vor. Zusätzlich wird nun eine abstrakte Klasse VirtualNetwork abgeleitet. Verfeinerungen davon stellen die Klassen CAE_LIS, CAE_VLAN und CAE_ELAN dar, wie Abbildung 4.6 zeigt. Instanzen dieser Klassen repräsentieren jeweils *ein* spezielles virtuelles Netz, nämlich ein LIS, ein (allgemeines) VLAN oder ein ELAN. Durch diesen Gruppierungsmechnismus können virtuelle Netze als Einheit betrachtet werden. Ein weiterer denkbarer Typ für virtuelle Netze wären beispielsweise *Virtual Private Networks (VPNs)*.

Objekte der Klassen CAE_VLAN und CAE_ELAN können über den Wert des Attributes VLANID bzw. ELANName eindeutig identifiziert werden. Für Objekte der Klasse CAE_LIS muß ein eindeutiger Wert für das ererbte Attribut Name gewählt werden.

Das Attribut ELANType von CAE_ELAN kann die Werte `802.3` (=1) oder `802.5` (=2) erhalten, je nachdem, ob ein Ethernet- oder Token Ring-LAN emuliert wird. Nur LECEndpoint-Objekte gleichen Typs können an der Aggregationsbeziehung InELAN teilhaben. Die Multiplizität dieser Beziehung auf Seiten von CAE_ELAN ist 0..1, da ein LEC in einem ELAN sein kann, aber nicht sein *muß*, beispielsweise vor Abschluß der *join*-Phase, vgl. [LANE 97]. Ähnliches gilt in analoger Weise für die übrigen in Abbildung 4.6 dargestellten Aggregationsbeziehungen.

Zum Verständnis der InVLAN-Beziehung ist folgendes anzumerken: CIM_VLAN stellt einen *Service Access Point* dar, CAE_VLAN dagegen ein *Logical Network*. Objekte der Klasse CIM_VLAN entsprechen VLANs, wie sie auf einzelnen LAN-Switches gemäß dem

VLAN-Teilmodell (vgl. Abschnitt 4.1.2) eingerichtet werden können. Ein Objekt der Klasse CAE_VLAN dagegen modelliert auch VLANs, die über mehrere Switches verteilt und eventuell über ein ATM-Backbone hinweg gebildet werden. In letzterem Fall erfolgt die Kopplung der LAN-Switches mit Hilfe der LAN Emulation und entsprechenden LECs auf den LAN-Switches. Dies kann durch die bereits vorgestellten Klassen und Beziehungen modelliert werden, da ein LECEndpoint eine Verfeinerung von LANEndpoint bildet. Damit können auch Instanzen der Switchable-Beziehung gebildet werden, die den LEC einem (logischen) Port des LAN-Switches zuordnen (Instanz der Klasse SwitchPort). Dieser wiederum kann einem bestimmten VLAN zugewiesen werden.

Dienste zur Realisierung virtueller Netze

Alle beschriebenen Typen virtueller Netze haben gemeinsam, daß für ihre Bereitstellung gesonderte Funktionalität vorhanden sein muß. Dies drückt die Klasse VirtualNetwork-Service aus, die im UML-Diagramm von Abbildung 4.7 dargestellt ist. Davon werden die Klassen VLANService, ELANService und LISService abgeleitet. Instanzen dieser Klassen stellen den Sammelpunkt dar, um alle für die Realisierung eines bestimmten LIS, ELAN oder VLAN benutzen (Client- oder Server-)Dienste zusammenzufassen, wie Abbildung 4.8 am Beispiel der Klasse ELANService zeigt. Die Multiplizitäten der Beziehungen ergeben sich direkt aus der LANE Spezifikation, beispielsweise ist ein LECS nicht zwingend erforderlich (vorkonfigurierter LES), ein LES kann nur ein ELAN bedienen, aber die LE Clients in einem ELAN verschiedene LE Server benutzen. Solche Beziehungen unter den einzelnen Server-Diensten können von der ServiceServiceDependency abgeleitet werden. Ein Beispiel dafür ist LES_BUS_pair, die den BUS angibt, der zu einem LES gehört.

Modellierung generischer Client/Server–Dienste

LANE und CLIP haben gemeinsam, daß sie über (verteilte) Client/Server–Dienste realisiert werden. Für das Management solcher Dienste wurden in [Guts 95] und [Muel 98] generische Objektmodelle entwickelt. Abbildung 4.9 zeigt eine Anpassung der dort verwendeten Klassen compObject (aus dem RM-ODP), Server und Client und ordnet sie in das CIM Schema ein. Von diesen generischen Klassen werden Client- und Server-Klassen für LANE und CLIP abgeleitet. Mit der ServiceAccessBySAP-Beziehung können die Dienste mit den ATM-Schnittstellen assoziiert werden, über die sie erreichbar sind.

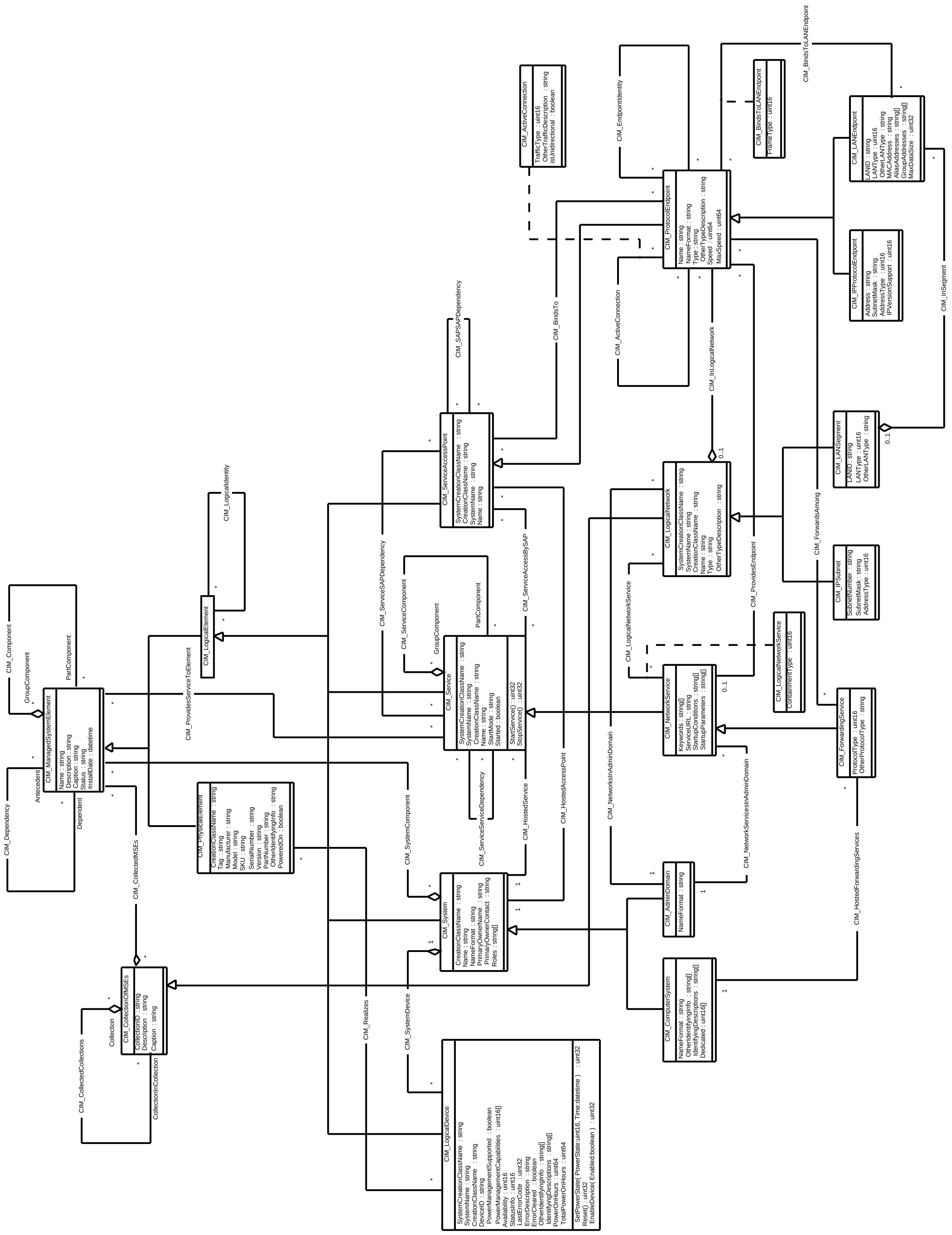

Abbildung 4.2: UML-Diagramm des reduzierten CIM Network Schemas

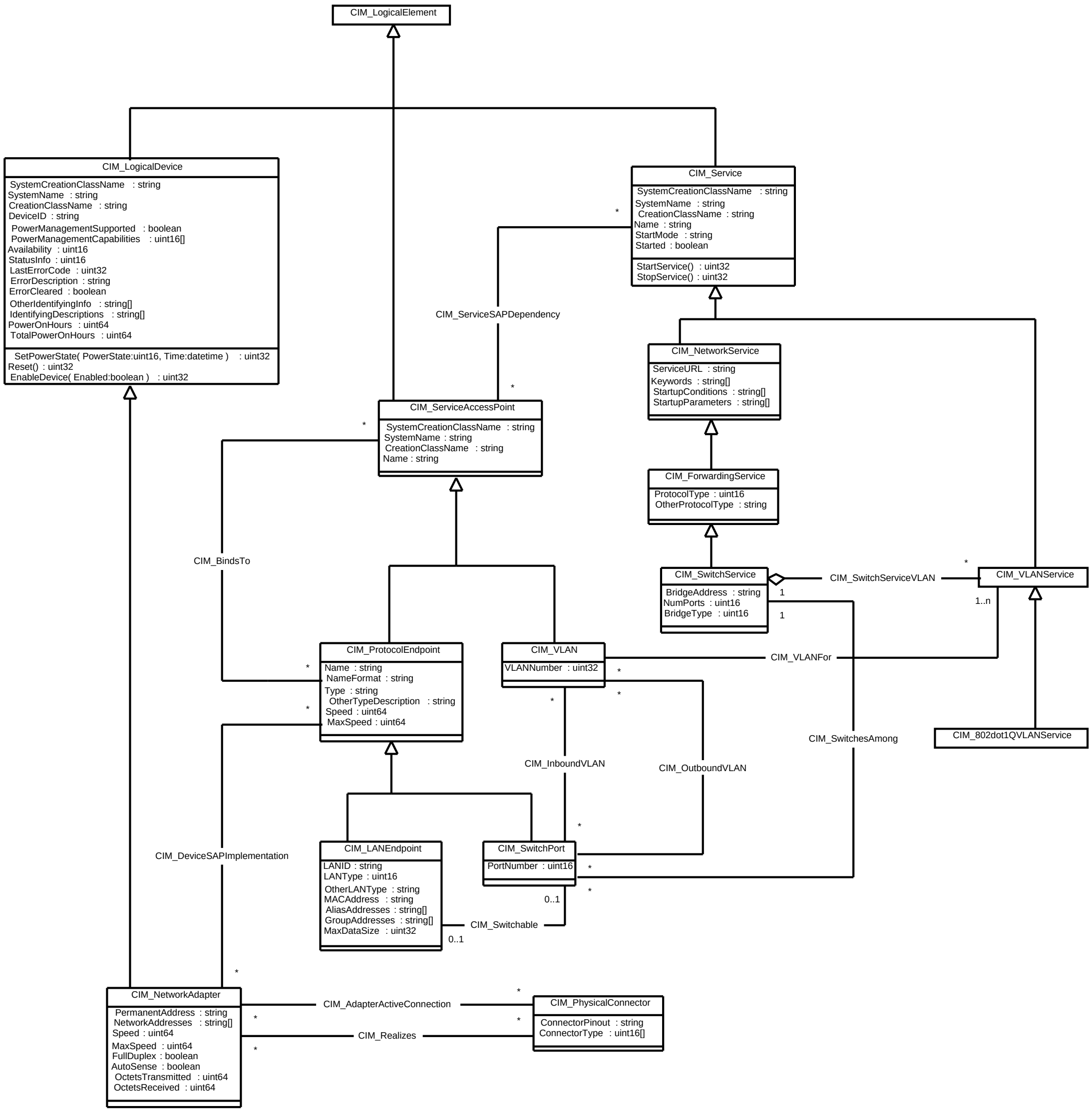

Abbildung 4.3: UML-Diagramm des reduzierten VLAN-Submodells

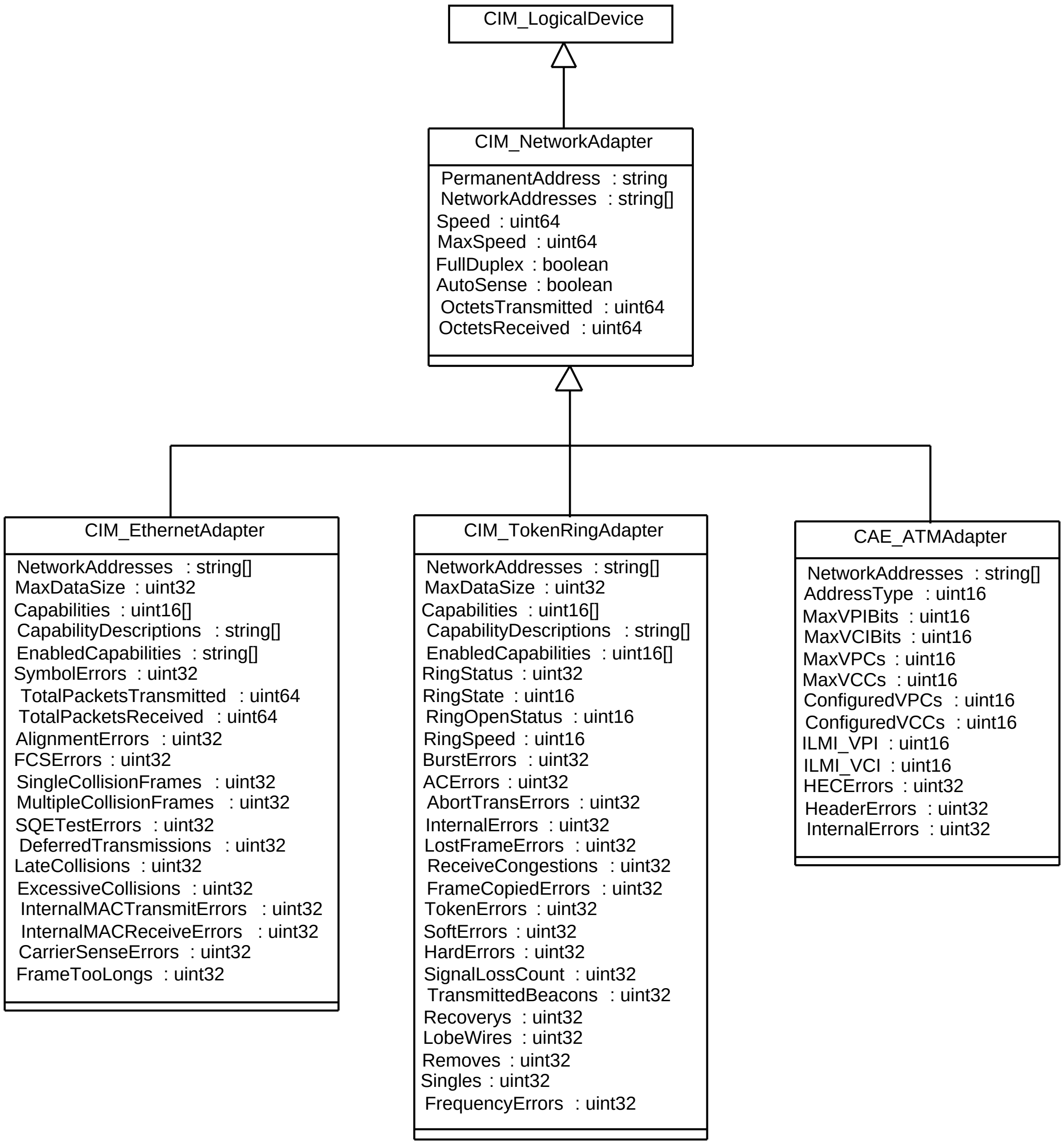

Abbildung 4.4: Modellierung einer ATM-Karte durch Verfeinerung

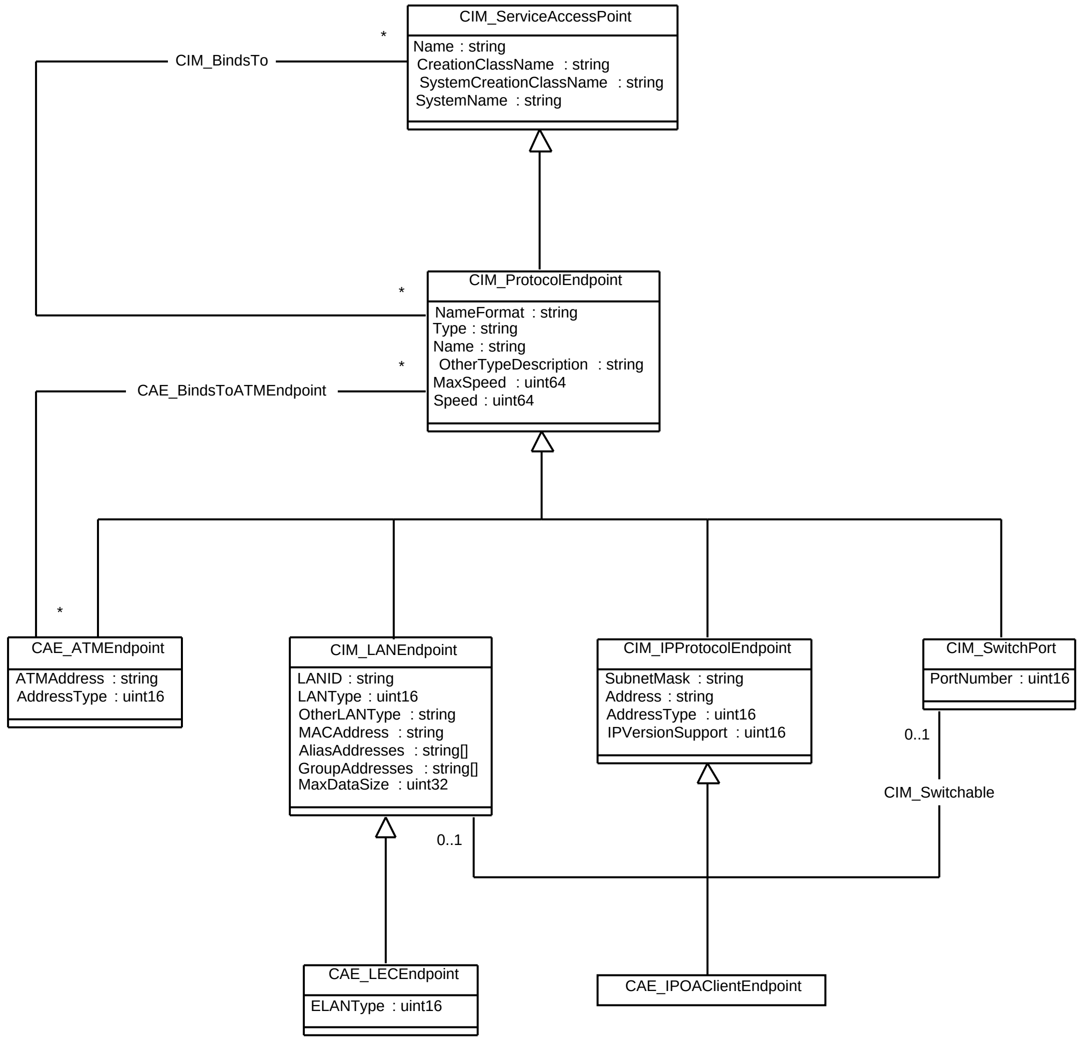

Abbildung 4.5: Ableitung ATM-spezifischer Protocol Endpoints

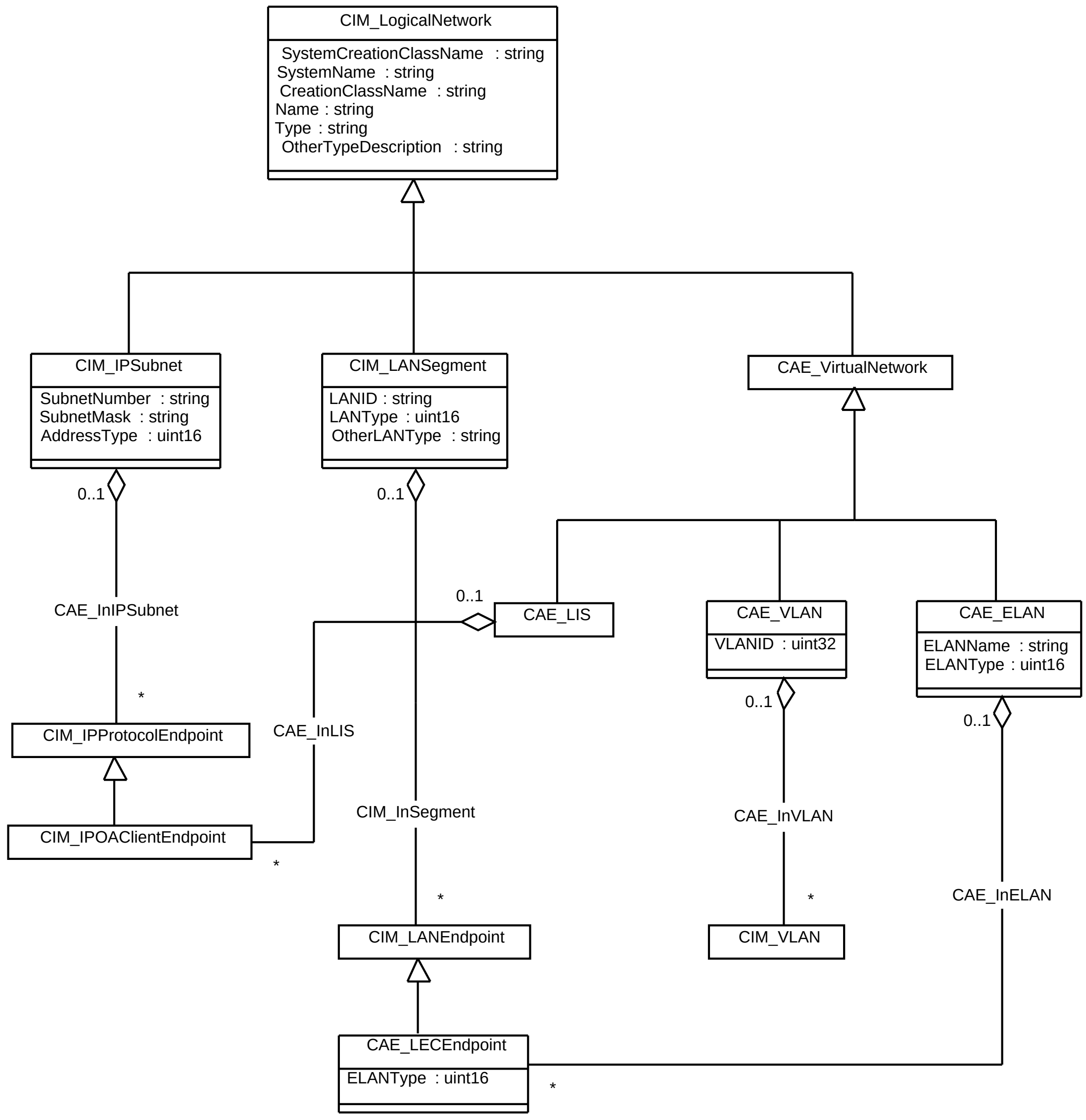

Abbildung 4.6: Gruppierung von Protocol Endpoints zu Logical Networks

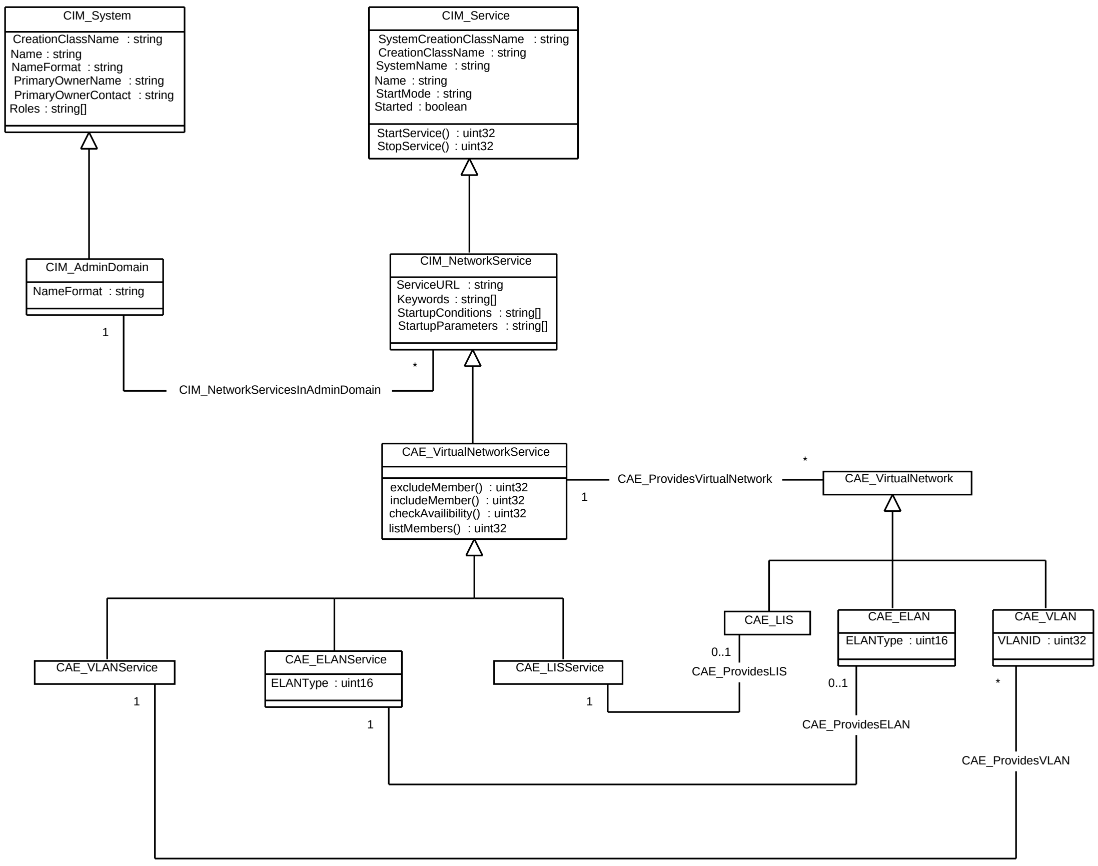

Abbildung 4.7: Verfeinerung der Klasse CIM_NetworkService

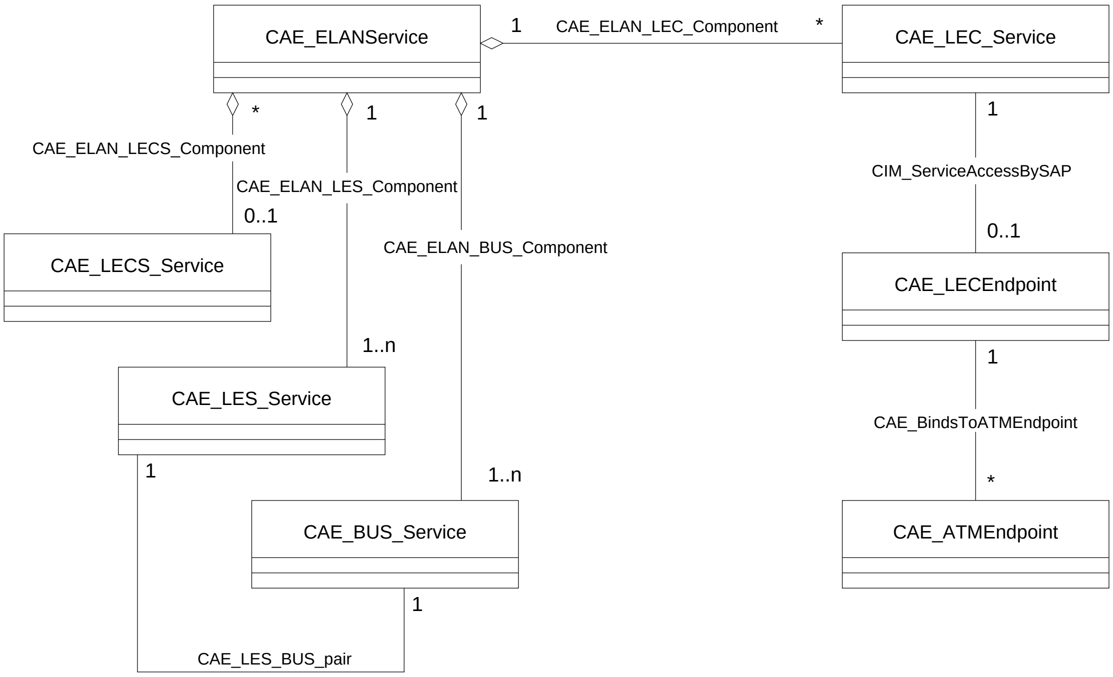

Abbildung 4.8: Komponenten eines ELANService

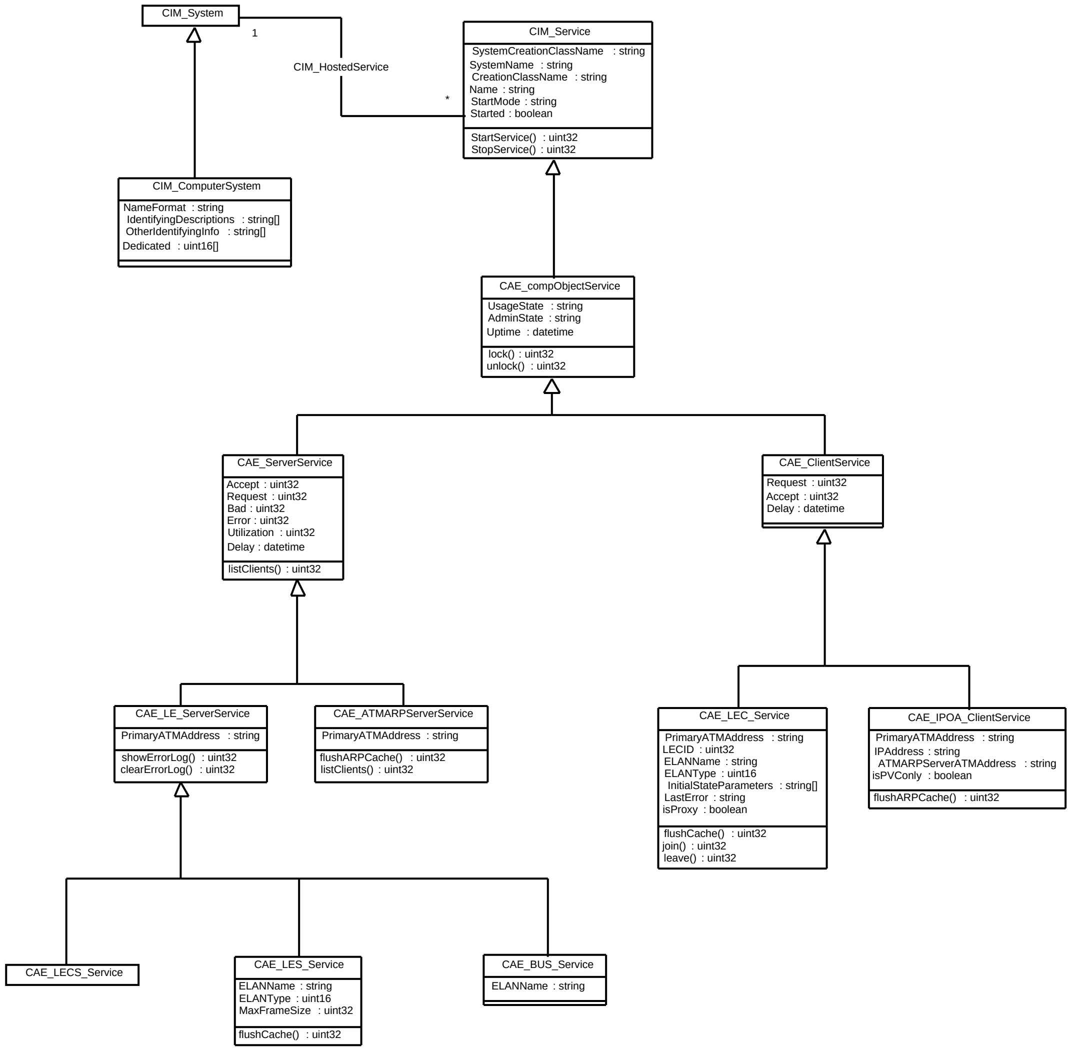

Abbildung 4.9: Modellierung generischer Client/Server–Dienste

Kapitel 5

Ausblick

Die ursprünglich geplante Entwicklung einer prototypisch implementierten Managementanwendung auf Basis der vorgestellten Modelle konnte im Rahmen dieser Arbeit nicht verwirklicht werden. Da CIM nur ein Informationsmodell vorgibt, stünde hier zunächst die Wahl einer geeigneten Managementarchitektur zur Debatte. Eine vielversprechende Lösung, die am Lehrstuhl entwickelt wurde, stellt hier das Java/CORBA-basierte Agentensystem *MASA* [Kemp 98, MASA 99] dar. Einen ersten Schritt auf dem Weg zu einem CORBA-basierten Agenten für die CIM Managementobjektklassen stellt die Gewinnung der Schnittstellenbeschreibungen für die Klassen in der *Interface Definition Language* (IDL) dar.

Das CASE-Tool *Software through Pictures* [StP 96a], das für die UML-Modellierung der CIM-Modelle eingesetzt wurde (siehe Abbildung 5.1), beherrscht eine automatische Generierung von IDL-Beschreibungen auf Grundlage von *Class Table*-Definitionen. Tabelle 5.1 gibt dazu eine mögliche Abbildung der Datentypen zwischen CIM und IDL sowie zu Java als möglicher Implementierungssprache vor. In Anhang C ist der generierte IDL-Code wiedergegeben. Zu beachten ist, daß zu den Assoziationen des Modells kein Code erzeugt werden kann, da IDL eine reine Schnittstellen-Beschreibungssprache darstellt. Der IDL-Code kann im Anschluß als Grundlage für die Gewinnung von Java-Codegrüsten mit einem IDL-Compiler dienen.

CIM	**IDL (CORBA)**	**Java**
`boolean`	`boolean`	`boolean`
`uint16`	`unsigned short`	`short`
`uint32`	`unsigned long`	`int`
`uint64`	`unsigned long long`	`long`
`string`	`string`	`java.lang.string`
`datetime`	`string`	`java.lang.string`
`uint16[]`	`sequence<unsigned short>`	`short[]`
`string[]`	`sequence<string>`	`java.lang.string[]`

Tabelle 5.1: Abbildung von Datentypen zwischen CIM, IDL und Java

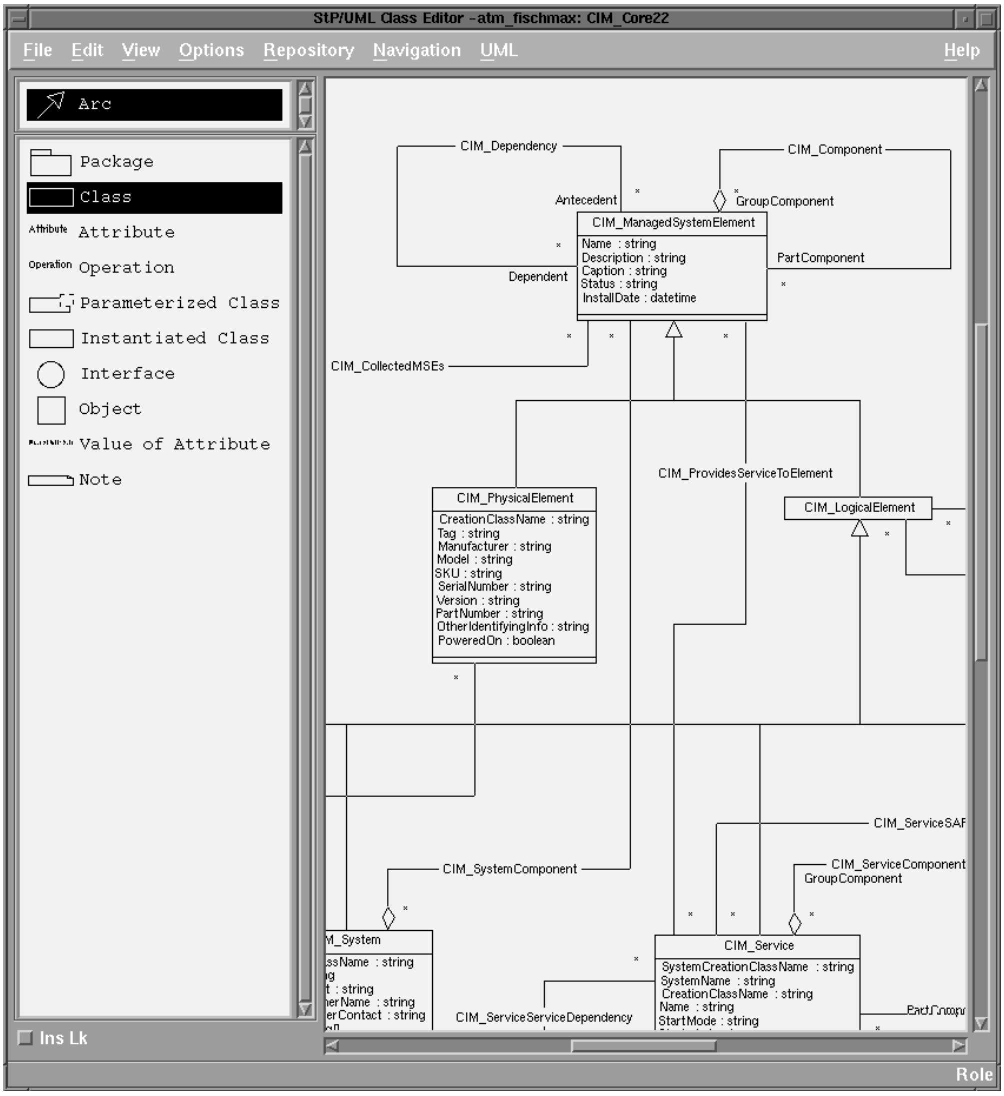

Abbildung 5.1: Der StP-Editor für Klassendiagramme

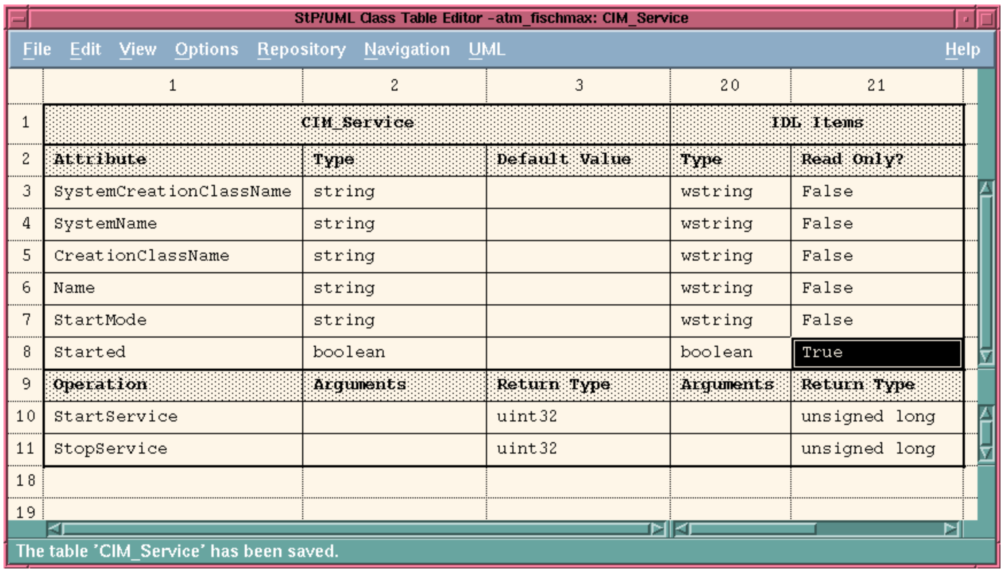

Abbildung 5.2: Erweiterung der *Class Table* Definition um IDL-Konstrukte

Anhang A

UML-Diagramme zum CIM

Auf den folgenden Seiten sind die UML-Diagramme zum CIM Core und Network Schema in der Version 2.2 [CIM 99b] abgebildet. Im einzelnen handelt es sich um:

- Seite 53: CIM Core Schema
- Seite 55: CIM Network Schema, BGP-Submodell
- Seite 57: CIM Network Schema, Multiprotocol-Bridge-Submodell
- Seite 59: CIM Network Schema, VLAN-Submodell
- Seite 61: CIM Network Schema, vereinfachte StP-Darstellung

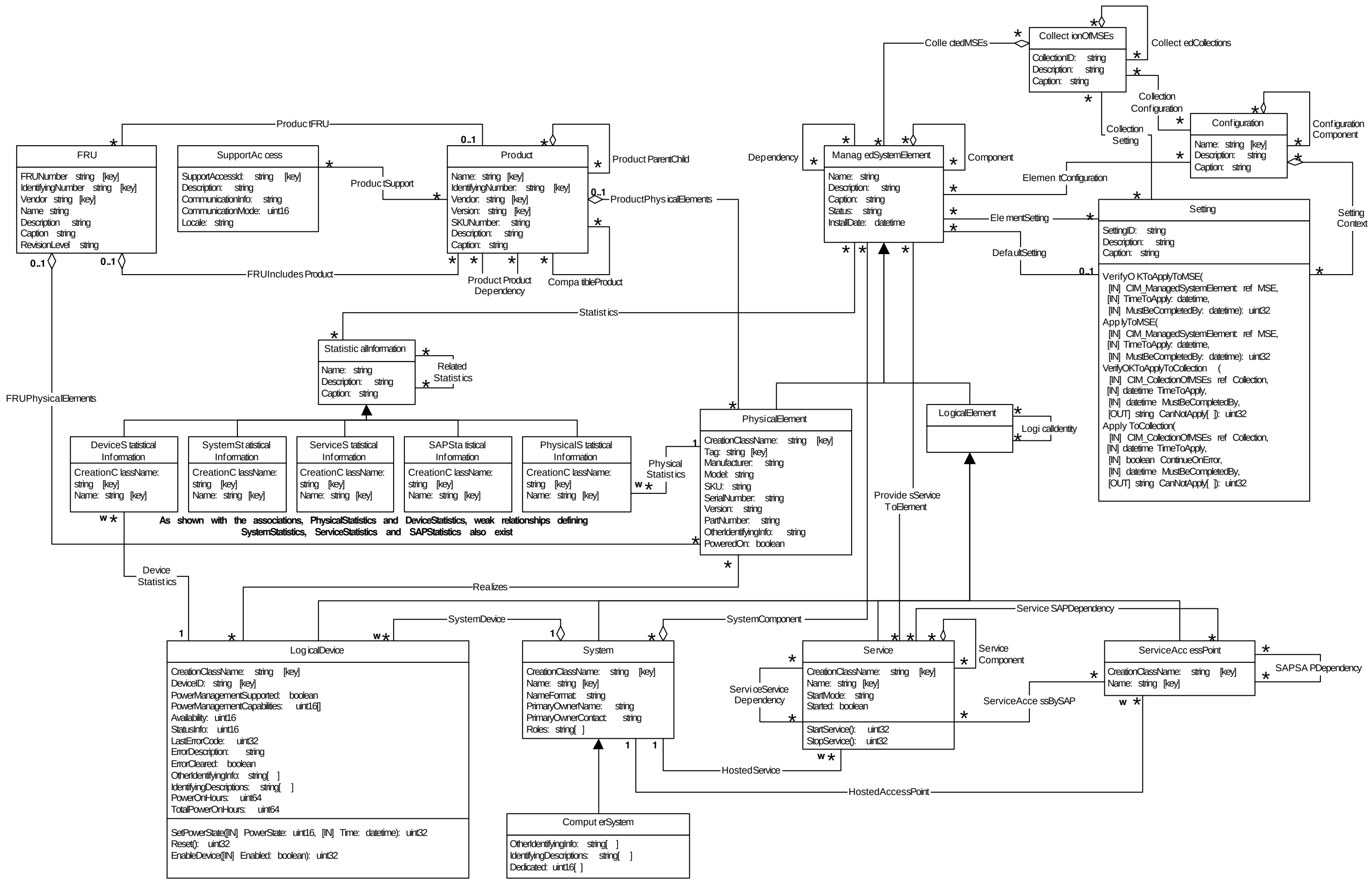

Abbildung A.1: CIM Core Schema

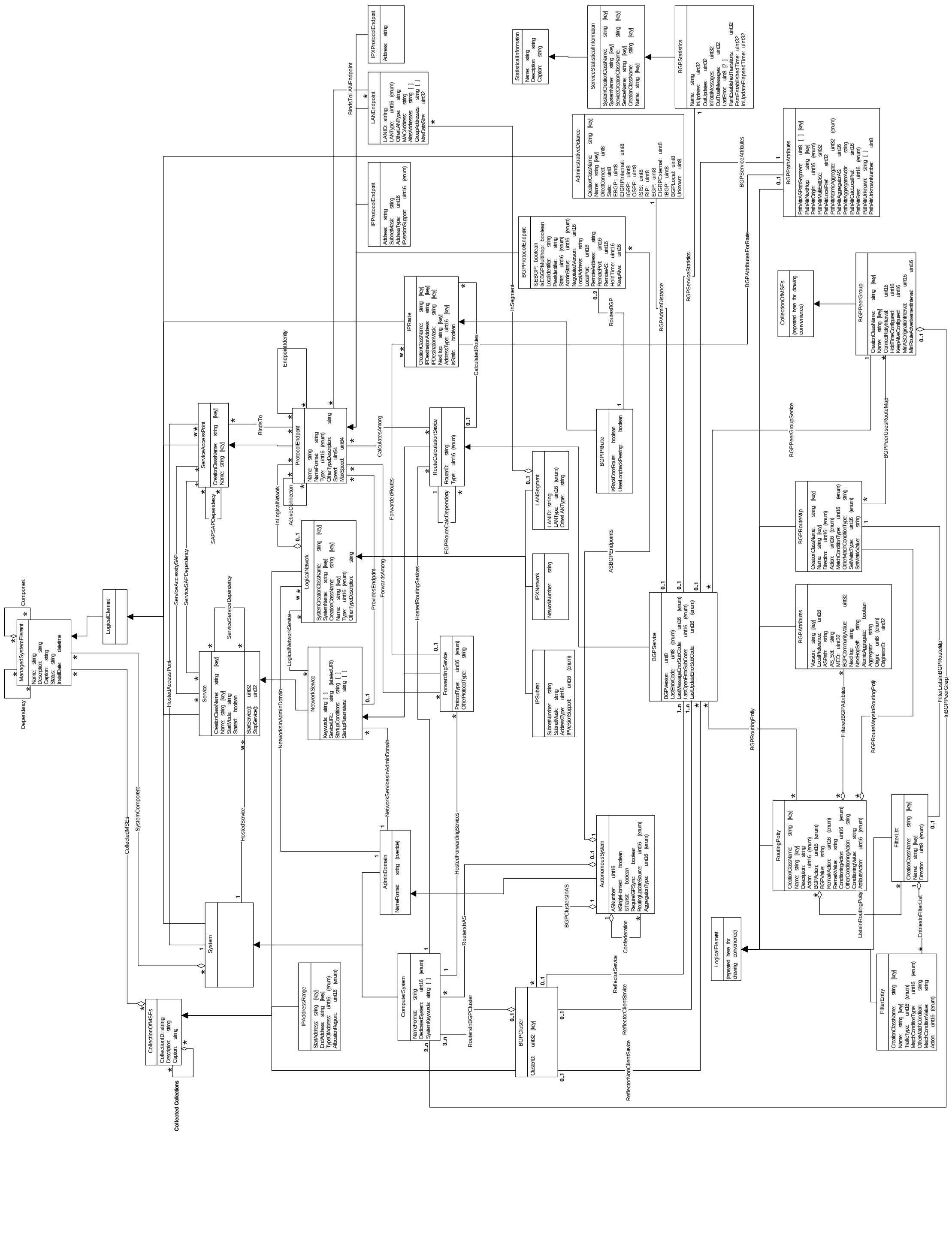

Abbildung A.2: BGP-Submodell im CIM Network Schema

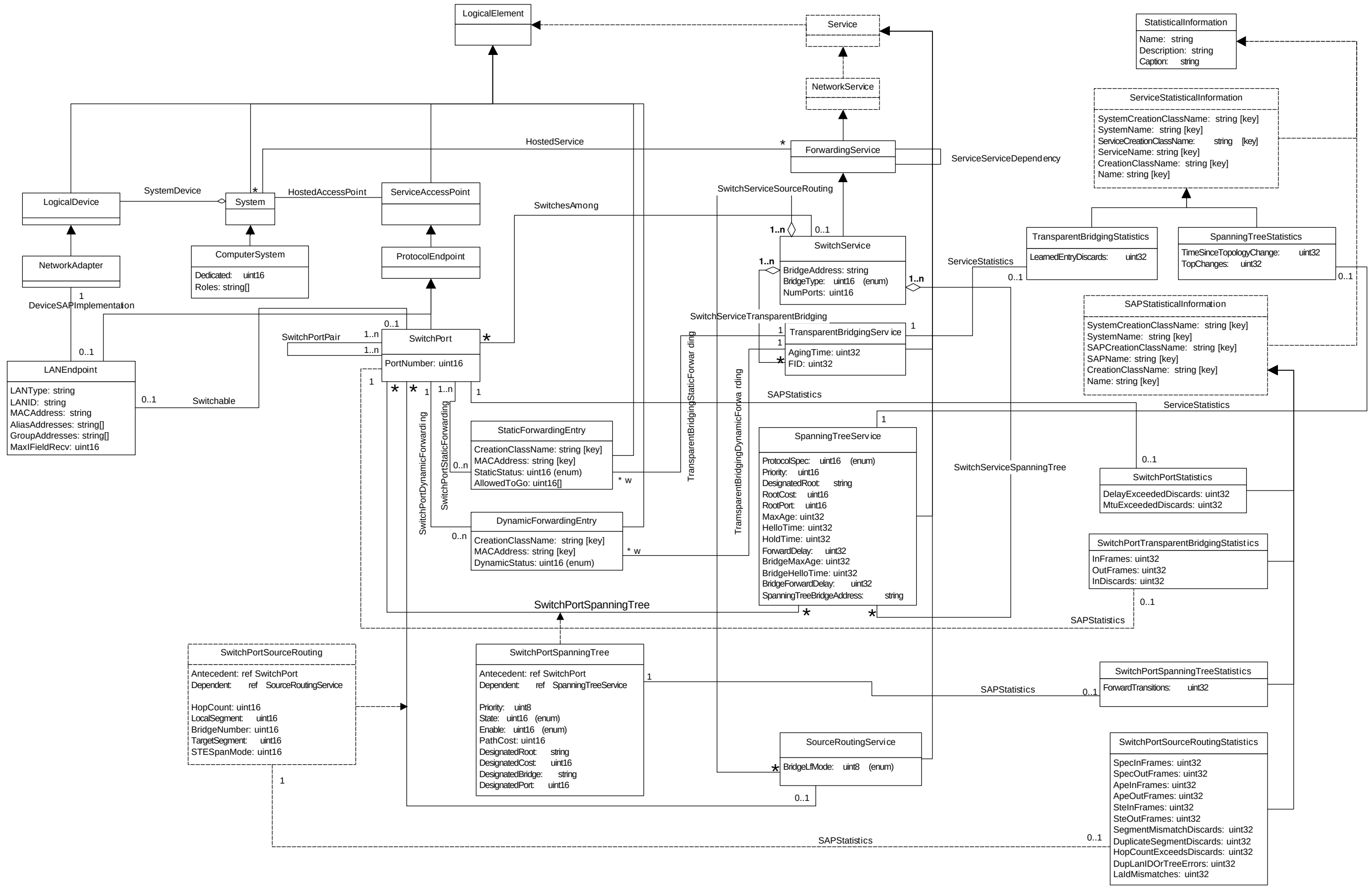

Abbildung A.3: Multiprotokoll-Bridge-Submodell im CIM Network Schema

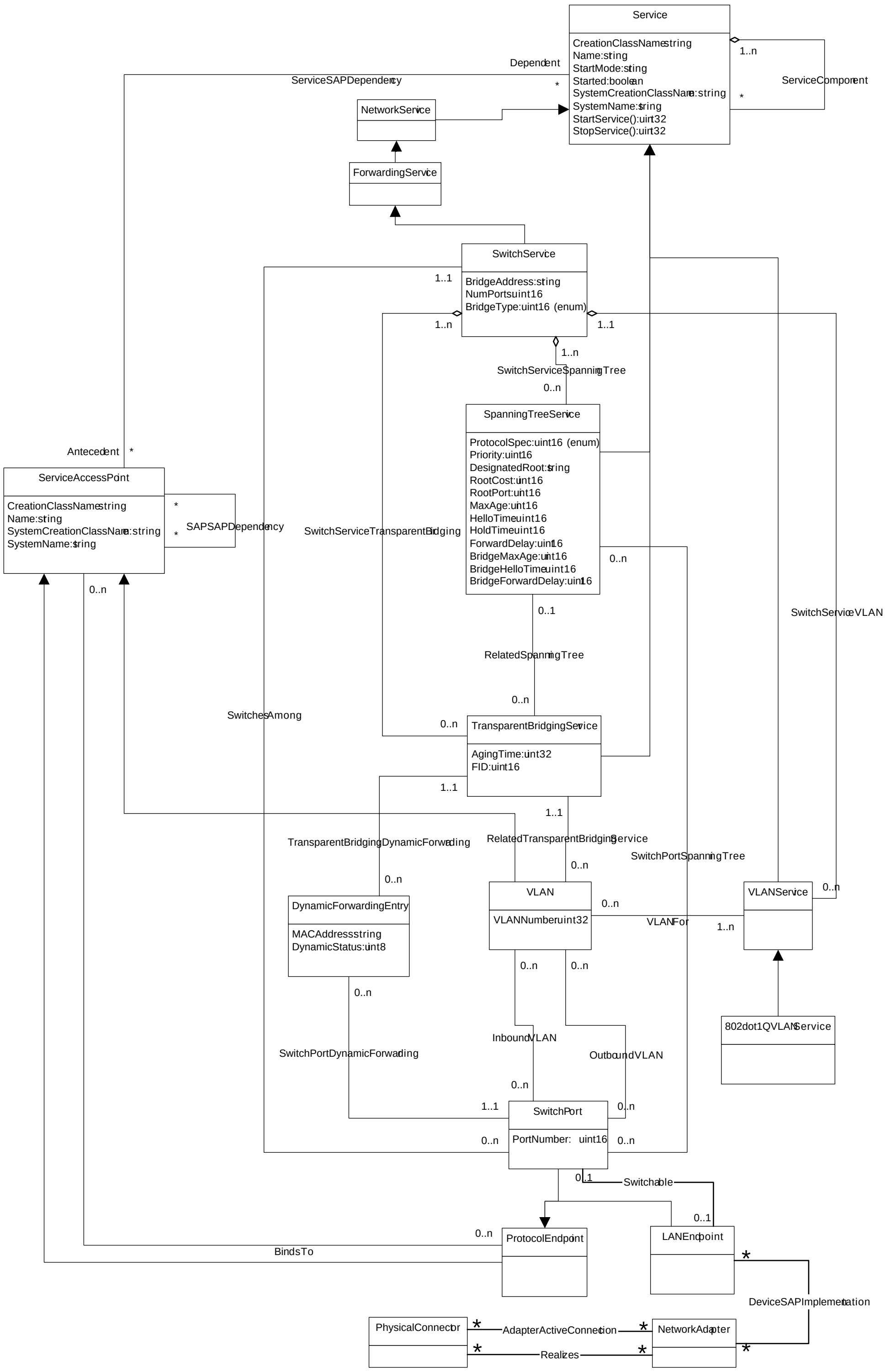

Abbildung A.4: VLAN-Submodell im CIM Network Schema

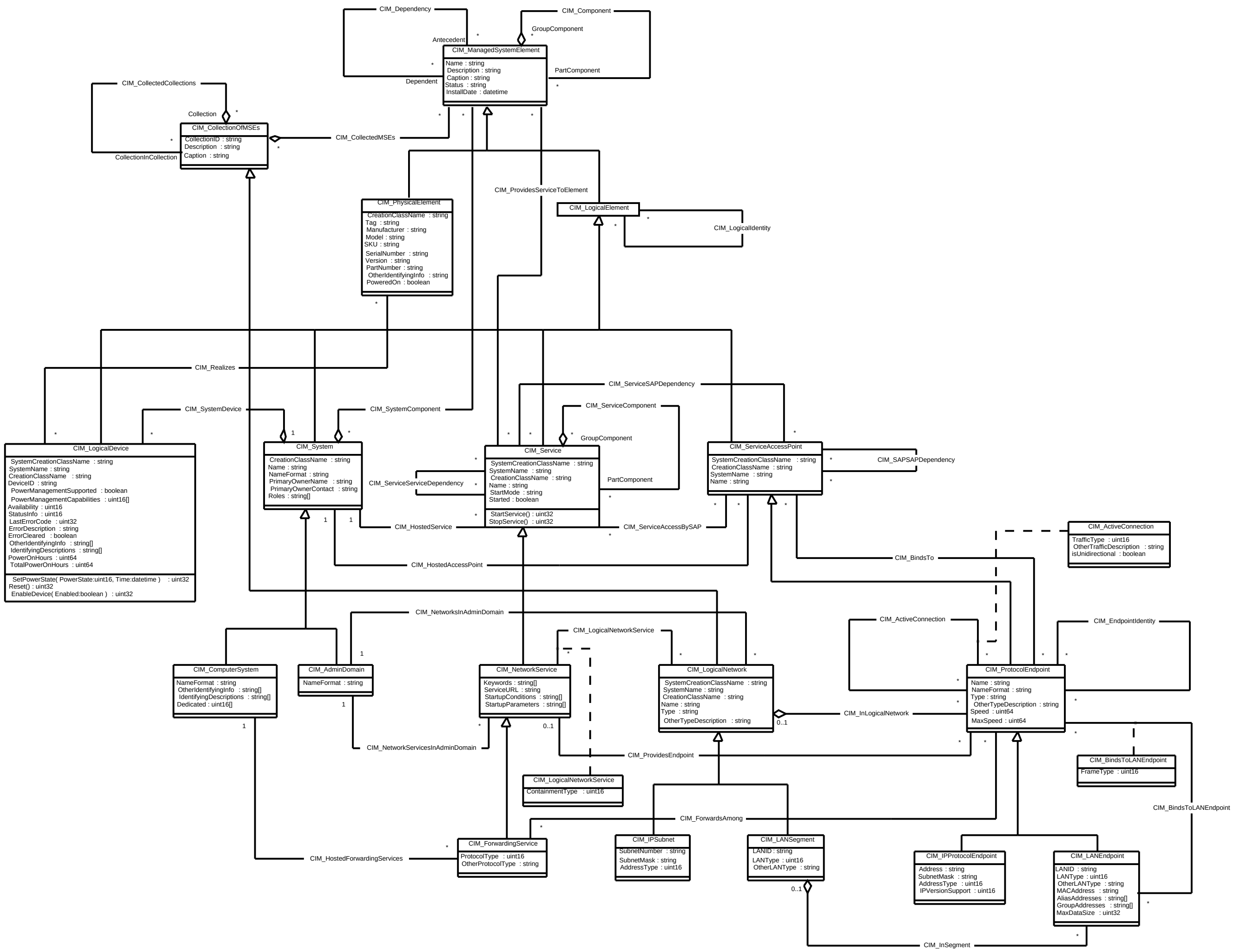

Abbildung A.5: UML-Diagramm des reduzierten CIM Network Schemas

Anhang B

MOF-Definition wichtiger CIM-Klassen

Die folgenden Abschnitte geben auszugsweise die Definitionen wesentlicher Klassen aus dem CIM Schema in der Version 2.2 wieder. Die vollständigen MOF-Dateien sind unter [CIM 99b] verfügbar.

B.1 CIM Core Schema

```
// ==================================================================
// Title:       Core MOF Specification 2.2
// Filename:    CIM_Core22.MOF
// Last Change: 06/08/1999
// ==================================================================

[...]

// ==================================================================
//    ManagedSystemElement
// ==================================================================
        [Abstract, Description (
         "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.") ]
class CIM_ManagedSystemElement
{
        [MaxLen (64), Description (
         "The Caption property is a short textual description (one-"
   "line string) of the object.") ]
    string Caption;
        [Description (
         "The Description property provides a textual description of "
         "the object.") ]
    string Description;
        [Description (
         "A datetime value indicating when the object was installed. "
         "A lack of a value does not indicate that the object is not "
         "installed."),
         MappingStrings {"MIF.DMTF|ComponentID|001.5"} ]
    datetime InstallDate;
        [MaxLen (256), Description (
         "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.") ]
    string Name;
  [MaxLen (10), Description (
         "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\" indicates that "
         "the Element is functioning, but needs attention. Examples "
         "of \"Stressed\" states are overload, overheated, etc. The "
         "condition \"Pred Fail\" (failure predicted) indicates that "
         "an Element is functioning properly but predicting a failure "
         "in the near future. An example is a SMART-enabled hard "
         "drive. Non-operational statuses can also be specified. These "
         "are \"Error\", \"NonRecover\", \"Starting\", \"Stopping\" "
         "and \"Service\". \"NonRecover\" indicates 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 one of "
         "the other states."),
         ValueMap {"OK", "Error", "Degraded", "Unknown", "Pred Fail",
             "Starting", "Stopping", "Service", "Stressed",
             "NonRecover"} ]
    string Status;
};

[...]

// ==================================================================
//    PhysicalElement
// ==================================================================
        [Abstract, Description (
         "Subclasses of CIM_PhysicalElement 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 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. These are tangible Managed "
         "System Elements (usually actual 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.") ]
class CIM_PhysicalElement:CIM_ManagedSystemElement
{
        [Key, MaxLen (256), Description (
         "CreationClassName indicates 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.") ]
    string CreationClassName;
        [MaxLen (256), Description (
         "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 "
         "CIM_Product.") ]
    string Manufacturer;
        [MaxLen (64), Description (
         "The name by which the PhysicalElement is generally known.") ]
    string Model;
        [MaxLen (64), Description (
         "The stock keeping unit number for this PhysicalElement.") ]
    string SKU;
        [MaxLen (64), Description (
         "A manufacturer-allocated number used to identify the Physical"
         "Element.") ]
    string SerialNumber;
```

```
        [Key, MaxLen (256), Description (
         "An arbitrary string that uniquely identifies the Physical"
   "Element 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 very high in the "
   "object hierarchy in order 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 continues to exist - and may even be inserted into a "
   "different scoping container.  Therefore, the key for Physical"
   "Element is an arbitrary string and is defined independently "
   "of any placement or location-oriented hierarchy.") ]
    string Tag;
        [MaxLen (64), Description (
   "A string indicating the version of the PhysicalElement.") ]
    string Version;
  [MaxLen (256), Description (
   "The part number assigned by the organization responsible for "
   "producing or manufacturing the PhysicalElement.") ]
    string PartNumber;
  [Description (
   "OtherIdentifyingInfo captures additional data, beyond asset "
   "tag information, that could be used to identify a Physical"
   "Element. 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.") ]
    string OtherIdentifyingInfo;
  [Description (
   "Boolean indicating that the PhysicalElement is powered on "
   "(TRUE), or is currently off (FALSE).") ]
    boolean PoweredOn;
};

// ==================================================================
//    LogicalElement
// ==================================================================
        [Abstract, Description (
         "CIM_LogicalElement is a base class for all the components of "
         "a System that represent abstract system components, such "
         "as Files, Processes, or system capabilities in the form "
         "of Logical Devices.") ]
class CIM_LogicalElement:CIM_ManagedSystemElement
{
};

[...]

// ==================================================================
//    System
// ==================================================================
        [Abstract, Description (
         "A CIM_System 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.") ]
class CIM_System:CIM_LogicalElement
{
        [Key, MaxLen (256), Description (
         "CreationClassName indicates 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.") ]
    string CreationClassName;
        [Key, MaxLen (256), Override ("Name"), Description (
         "The inherited Name serves as key of a System instance in "
         "an enterprise environment.") ]
    string Name;
        [MaxLen (64), Description (
         "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.") ]
    string NameFormat;
        [MaxLen (256), Description (
         "A string that provides information on how the primary system "
         "owner can be reached (e.g. phone number, email address, "
   "...)."),
         MappingStrings {"MIF.DMTF|General Information|001.3"} ]
    string PrimaryOwnerContact;
        [MaxLen (64), Description (
   "The name of the primary system owner."),
         MappingStrings {"MIF.DMTF|General Information|001.4"} ]
    string PrimaryOwnerName;
        [Description (
         "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'.") ]
    string Roles[];
};

// ==================================================================
//    ComputerSystem
// ==================================================================
  [Abstract, Description (
         "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).") ]
class CIM_ComputerSystem : CIM_System
{
```

```
        [Override ("NameFormat"), Description (
     "The ComputerSystem 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 is defined to create the ComputerSystem Name "
   "to attempt to always generate the same Name, independent of "
   "discovery protocol. This prevents inventory and management "
   "problems where the same asset or entity is discovered "
   "multiple times, but can not be resolved to a single object. "
   "Use of the heuristic is optional, but recommended.\n\n"
   "The NameFormat property identifies how the ComputerSystem "
   "Name is generated, using a heuristic. The heuristic is "
   "outlined, in detail, in the CIM V2 System Model spec. "
   "It assumes that the documented rules are traversed in order, "
   "to determine and assign a Name. The NameFormat Values "
   "list defines the precedence order for assigning the Computer"
   "System Name. Several rules do map to the same Value.\n\n"
   "Note that the ComputerSystem Name calculated using the "
   "heuristic is the System's key value. Other names can be "
   "assigned and used for the ComputerSystem, that better suit "
   "a business, using Aliases."),
         ValueMap {"Other", "IP", "Dial", "HID", "NWA", "HWA", "X25",
       "ISDN", "IPX", "DCC", "ICD", "E.164", "SNA", "OID/OSI"} ]
   string NameFormat;
        [MaxLen (256), ArrayType ("Indexed"),
         Description (
         "OtherIdentifyingInfo captures additional data, beyond "
         "System Name information, that could be used to identify "
         "a ComputerSystem. One example would be to hold the "
         "Fibre Channel World-Wide Name (WWN) of a node. Note that "
         "if only the Fibre Channel name is available and is "
         "unique (able to be used as the System key), then this "
         "property would be NULL and the WWN would become the "
         "System key, its data placed in the Name property."),
         ModelCorrespondence {
          "CIM_ComputerSystem.IdentifyingDescriptions"} ]
   string OtherIdentifyingInfo[];
        [ArrayType ("Indexed"), Description (
         "An array of free-form strings providing explanations "
         "and details behind the entries in the OtherIdentifying"
         "Info array. Note, each entry of this array is related "
         "to the entry in OtherIdentifyingInfo that is located at "
         "the same index."),
         ModelCorrespondence {
          "CIM_ComputerSystem.OtherIdentifyingInfo"} ]
   string IdentifyingDescriptions[];
        [Description (
         "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 {"Not Dedicated", "Unknown", "Other", "Storage",
          "Router", "Switch", "Layer 3 Switch",
          "Central Office Switch", "Hub", "Access Server",
          "Firewall", "Print", "I/O", "Web Caching"} ]
   uint16 Dedicated[];
};
```

```
// ==================================================================
//    LogicalDevice
// ==================================================================
        [Abstract, Description (
         "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.") ]
class CIM_LogicalDevice:CIM_LogicalElement
{
        [Propagated("CIM_System.CreationClassName"),
         Key, MaxLen (256),
         Description ("The scoping System's CreationClassName.") ]
   string SystemCreationClassName;
        [Propagated("CIM_System.Name"),
         Key, MaxLen (256),
         Description ("The scoping System's Name.") ]
   string SystemName;
        [Key, MaxLen (256),
         Description (
         "CreationClassName indicates 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.") ]
   string CreationClassName;
        [Key, MaxLen (64),
         Description (
         "An address or other identifying information to uniquely "
         "name the LogicalDevice.") ]
   string DeviceID;
        [Description (
         "Boolean indicating that the Device can be power managed "
         "- ie, put into a power save state. This boolean does not "
         "indicate 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.") ]
   boolean PowerManagementSupported;
        [Description(
         "Indicates 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\" indicates 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) indicates that "
 "the SetPowerState method is supported. \"Power Cycling "
       "Supported\" (6) indicates that the SetPowerState method "
 "can be invoked with the PowerState input variable set to 5 ("
 "\"Power Cycle\"). \"Timed Power On Supported\" (7) indicates "
 "that the SetPowerState method can be invoked with the Power"
 "State input variable set to 5 (\"Power Cycle\") and the Time "
 "parameter set to a specific date and time, or interval, "
 "for power-on."),
       Values {"Unknown", "Not Supported", "Disabled",
     "Enabled", "Power Saving Modes Entered Automatically",
     "Power State Settable", "Power Cycling Supported",
     "Timed Power On Supported"} ]
 uint16 PowerManagementCapabilities[];
      [Description (
       "The availability and status of the Device.  For example, the "
       "Availability property indicates 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\") indicates "
 "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\") indicates 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\") "
 "indicates that the Device is in a warning state, though also "
 "in a power save mode."),
 ValueMap {"1", "2", "3", "4", "5", "6", "7", "8", "9",
     "10", "11", "12", "13", "14", "15", "16", "17", "18", "19"},
       Values {"Other", "Unknown", "Running/Full Power", "Warning",
     "In Test", "Not Applicable", "Power Off", "Off Line",
     "Off Duty", "Degraded", "Not Installed", "Install Error",
     "Power Save - Unknown", "Power Save - Low Power Mode",
     "Power Save - Standby", "Power Cycle",
     "Power Save - Warning", "Paused", "Not Ready"},
       MappingStrings {"MIF.DMTF|Operational State|003.5",
        "MIB.IETF|HOST-RESOURCES-MIB.hrDeviceStatus"} ]
 uint16 Availability;
      [Description (
       "StatusInfo is a string indicating whether the Logical"
       "Device 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."),
       ValueMap {"1", "2", "3", "4", "5"},
       Values {"Other", "Unknown", "Enabled",
     "Disabled", "Not Applicable"},
       MappingStrings {"MIF.DMTF|Operational State|003.3"} ]
 uint16 StatusInfo;
[Description (
 "LastErrorCode captures the last error code reported by "
 "the LogicalDevice.")]
 uint32 LastErrorCode;
[Description (
 "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.")]
 string ErrorDescription;
[Description (
 "ErrorCleared is a boolean property indicating that the "
 "error reported in LastErrorCode is now cleared.")]
 boolean ErrorCleared;
      [MaxLen (256), ArrayType ("Indexed"),
       Description (
       "OtherIdentifyingInfo captures additional data, beyond "
       "DeviceID information, that could be used to identify a "
       "LogicalDevice. One example would be to hold the Operating"
       "System's user friendly name for the Device in this "
       "property."),
       ModelCorrespondence {
         "CIM_LogicalDevice.IdentifyingDescriptions"} ]
 string OtherIdentifyingInfo[];
      [Description (
       "The number of consecutive hours that this Device has been "
       "powered, since its last power cycle."),
       Units ("Hours"), Counter ]
 uint64 PowerOnHours;
      [Description (
       "The total number of hours that this Device has been "
       "powered."),
       Units ("Hours"), Counter ]
 uint64 TotalPowerOnHours;
      [ArrayType ("Indexed"), Description (
       "An array of free-form strings providing explanations "
       "and details behind the entries in the OtherIdentifyingInfo "
       "array. Note, each entry of this array is related to the "
       "entry in OtherIdentifyingInfo that is located at the same "
       "index."),
       ModelCorrespondence {"CIM_LogicalDevice.OtherIdentifyingInfo"} ]
 string IdentifyingDescriptions[];
      [Description(
       "SetPowerState defines the desired power state for a "
       "LogicalDevice and when a Device should be put into that "
       "state. The desired power state is specified by setting "
 "the PowerState parameter to one of the "
       "following integer values: 1=\"Full Power\", 2=\"Power Save "
 "- Low Power Mode\", 3=\"Power Save - Standby\", 4=\"Power "
 "Save - Other\", 5=\"Power Cycle\" or 6=\"Power Off\". "
       "The Time parameter (for all state changes but 5, "
       "\"Power Cycle\") indicates when the power state should be "
       "set, either as a regular date-time value or as an interval "
       "value (where the interval begins when the method "
       "invocation is received). When the PowerState parameter is "
       "equal to 5, \"Power Cycle\", the Time parameter indicates "
 "when the Device should power on again. Power off is "
 "immediate. SetPowerState should return 0 if successful, 1 "
 "if the specified PowerState and Time request is not "
 "supported, and some other value if any other error "
 "occurred. In a subclass, the set of possible return codes "
       "could be specified, using a ValueMap qualifier on the method. "
       "The strings to which the ValueMap contents are 'translated' "
       "may also be specified in the subclass as a Values array "
       "qualifier.") ]
```

```
   uint32 SetPowerState([IN, ValueMap {"1", "2", "3", "4", "5", "6"},
    Values {"Full Power", "Power Save - Low Power Mode",
    "Power Save - Standby", "Power Save - Other", "Power Cycle",
    "Power Off"}] uint16 PowerState, [IN] datetime Time);
        [Description (
         "Requests a reset of the LogicalDevice. The return value "
         "should be 0 if the request was successfully executed, "
         "1 if the request is not supported and some other value "
         "if an error occurred. In a subclass, the set of possible "
         "return codes could be specified, using a ValueMap qualifier "
         "on the method. The strings to which the ValueMap contents "
         "are 'translated' may also be specified in the subclass as a "
         "Values array qualifier.") ]
   uint32 Reset();
        [Description (
         "Requests that the LogicalDevice be enabled (\"Enabled\" "
         "input parameter = TRUE) or disabled (= FALSE). If "
         "successful, the Device's StatusInfo property should also "
         "reflect the desired state (enabled/disabled). The return "
         "code should be 0 if the request was successfully executed, "
         "1 if the request is not supported and some other value if "
         "an error occurred. In a subclass, the set of possible "
         "return codes could be specified, using a ValueMap qualifier "
         "on the method. The strings to which the ValueMap contents "
         "are 'translated' may also be specified in the subclass as a "
         "Values array qualifier.") ]
   uint32 EnableDevice([IN] boolean Enabled);
};

// ==================================================================
//    Service
// ==================================================================
        [Abstract, Description (
         "A CIM_Service 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.") ]
class CIM_Service:CIM_LogicalElement
{
        [Key, MaxLen (256), Description (
   "CreationClassName indicates 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.") ]
    string CreationClassName;
        [Override ("Name"), Key, MaxLen (256),
         Description (
         "The Name property uniquely identifies the Service and "
         "provides an indication of the functionality that is "
         "managed. This functionality is described in more detail in "
         "the object's Description property. ") ]
    string Name;
        [MaxLen (10), Description (
         "StartMode is a string value indicating whether the Service "
         "is automatically started by a System, Operating System, etc. "
         "or only started upon request."),
   ValueMap {"Automatic", "Manual"} ]
    string StartMode;
        [Description (
         "Started is a boolean indicating whether the Service "
         "has been started (TRUE), or stopped (FALSE).") ]
    boolean Started;
        [Propagated ("CIM_System.CreationClassName"), Key,
   MaxLen (256), Description (
         "The scoping System's CreationClassName. ") ]
    string SystemCreationClassName;
        [Propagated ("CIM_System.Name"), Key, MaxLen (256),
         Description ("The scoping System's Name.") ]
    string SystemName;
        [Description (
         "The StartService method places the Service in the started "
         "state. It returns an integer value of 0 if the Service was "
         "successfully started, 1 if the request is not supported and "
         "any other number to indicate an error. In a subclass, the "
         "set of possible return codes could be specified, using a "
         "ValueMap qualifier on the method. The strings to which the "
         "ValueMap contents are 'translated' may also be specified in "
         "the subclass as a Values array qualifier.") ]
    uint32 StartService();
        [Description (
         "The StopService method places the Service in the stopped "
         "state. It returns an integer value of 0 if the Service was "
         "successfully stopped, 1 if the request is not supported and "
         "any other number to indicate an error. In a subclass, the "
         "set of possible return codes could be specified, using a "
         "ValueMap qualifier on the method. The strings to which the "
         "ValueMap contents are 'translated' may also be specified in "
         "the subclass as a Values array qualifier.") ]
    uint32 StopService();
};

// ==================================================================
//    ServiceAccessPoint
// ==================================================================
        [Abstract, Description (
         "CIM_ServiceAccessPoint represents the ability to utilize or "
         "invoke a Service.  Access points represent that a Service is "
         "made available to other entities for use.") ]
class CIM_ServiceAccessPoint:CIM_LogicalElement
{
        [Key, MaxLen (256), Description (
   "CreationClassName indicates 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.") ]
    string CreationClassName;
        [Override ("Name"), Key, MaxLen (256),
         Description (
         "The Name property uniquely identifies the ServiceAccessPoint "
         "and provides an indication of the functionality that is "
         "managed.  This functionality is described in more detail in "
         "the object's Description property.") ]
```

```
    string Name;
        [Propagated ("CIM_System.CreationClassName"), Key,
   MaxLen (256), Description (
         "The scoping System's CreationClassName.") ]
    string SystemCreationClassName;
        [Propagated ("CIM_System.Name"), Key, MaxLen (256),
   Description ("The scoping System's Name.") ]
    string SystemName;
};

// ==================================================================
//    Dependency
// ==================================================================
        [Association, Abstract, Description (
         "CIM_Dependency is a generic association used to establish "
         "dependency relationships between objects.") ]
class CIM_Dependency
{
        [Key, Description (
         "Antecedent represents the independent object in this "
         "association.") ]
    CIM_ManagedSystemElement REF Antecedent;
        [Key, Description (
         "Dependent represents the object dependent on the "
         "Antecedent.") ]
    CIM_ManagedSystemElement REF Dependent;
};

[...]

// ==================================================================
//    HostedService
// ==================================================================
        [Association, Description (
         "CIM_HostedService is an association between a Service and "
         "the System on which the functionality resides.  The "
         "cardinality of this association is 1-to-many.  A System may "
         "host many Services. Services are weak with respect to their "
         "hosting System. Heuristic:  A Service is hosted on the "
         "System where the LogicalDevices or SoftwareFeatures that "
         "implement the Service are located.  The model does not "
         "represent Services hosted across multiple systems.  This is "
         "modeled as an ApplicationSystem that acts as an aggregation "
         "point for Services, that are each located on a single "
         "host.") ]
class CIM_HostedService:CIM_Dependency
{
        [Override ("Antecedent"), Max (1), Min (1),
   Description ("The hosting System.") ]
    CIM_System REF Antecedent;
        [Override ("Dependent"), Weak,
         Description ("The Service hosted on the System.") ]
    CIM_Service REF Dependent;
};

// ==================================================================
//    HostedAccessPoint
// ==================================================================
        [Association, Description (
         "CIM_HostedAccessPoint is an association between a Service"
         "AccessPoint and the System on which it is provided.  The "
         "cardinality of this association is 1-to-many and is weak "
         "with respect to the System. Each System may host many "
   "ServiceAccessPoints.  Heuristic:  If the implementation of "
         "the ServiceAccessPoint is modeled, it must be "
         "implemented by a Device or SoftwareFeature that is part of "
         "the System hosting the ServiceAccessPoint.") ]
class CIM_HostedAccessPoint:CIM_Dependency
{
        [Override ("Antecedent"), Max (1), Min (1),
   Description ("The hosting System.") ]
    CIM_System REF Antecedent;
        [Override ("Dependent"), Weak,
 Description ("The SAP(s) that are hosted on this System.") ]
    CIM_ServiceAccessPoint REF Dependent;
};

// ==================================================================
//    ServiceServiceDependency
// ==================================================================
        [Association, Description (
         "CIM_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 "
         "upon 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 utilize the "
   "SAPs of this Service.  This usage dependency is modeled via "
         "the CIM_ServiceSAPDependency association.") ]
class CIM_ServiceServiceDependency:CIM_Dependency
{
        [Override ("Antecedent"),
   Description ("The required Service.") ]
    CIM_Service REF Antecedent;
        [Override ("Dependent"), Description (
         "The Service that is dependent on an underlying Service.") ]
    CIM_Service REF Dependent;
  [Description (
         "The nature of 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 {"Unknown", "Other", "Service Must Have Completed",
         "Service Must Be Started",
         "Service Must Not Be Started"} ]
    uint16 TypeOfDependency;
};

// ==================================================================
//    ServiceSAPDependency
// ==================================================================
        [Association, Description (
         "CIM_ServiceSAPDependency is an association between a Service "
```

```
         "and a ServiceAccessPoint indicating that the referenced SAP "
         "is utilized by the Service to provide its functionality. For "
         "example, Boot Services may invoke BIOS' Disk Services "
         "(interrupts) in order to function.") ]
class CIM_ServiceSAPDependency:CIM_Dependency
{
        [Override ("Antecedent"),
   Description ("The required ServiceAccessPoint") ]
    CIM_ServiceAccessPoint REF Antecedent;
        [Override ("Dependent"), Description (
         "The Service that is dependent on an underlying SAP.") ]
    CIM_Service REF Dependent;
};

// ==================================================================
//    SAPSAPDependency
// ==================================================================
        [Association, Description (
         "CIM_SAPSAPDependency is an association between a Service"
         "AccessPoint and another ServiceAccessPoint indicating that "
         "the latter is required in order for the former ServiceAccess"
         "Point to utilize or connect with its Service. For example, "
         "to print at a network printer, local Print Access Points "
         "must utilize underlying network-related SAPs, or "
         "ProtocolEndpoints, in order to send the print request.") ]
class CIM_SAPSAPDependency:CIM_Dependency
{
        [Override ("Antecedent"), Description (
         "The required ServiceAccessPoint.") ]
    CIM_ServiceAccessPoint REF Antecedent;
        [Override ("Dependent"), Description (
         "The ServiceAccessPoint that is dependent on an underlying "
         "SAP.") ]
    CIM_ServiceAccessPoint REF Dependent;
};

// ==================================================================
//    ProvidesServiceToElement
// ==================================================================
        [Association, Description (
         "CIM_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 AlertOn"
         "LAN Services to signal an incomplete or erroneous boot, and "
         "hardware-related errors.") ]
class CIM_ProvidesServiceToElement : CIM_Dependency
{
        [Override ("Antecedent"), Description (
         "The Service provided.") ]
   CIM_Service ref Antecedent;
        [Override ("Dependent"), Description (
         "The ManagedSystemElement dependent on the Service.") ]
   CIM_ManagedSystemElement ref Dependent;
};

// ==================================================================
//    Realizes
// ==================================================================
        [Association, Description (
         "CIM_Realizes is the association that defines the mapping "
         "between a Logical Device and the physical component that "
         "implements the Device.") ]
class CIM_Realizes:CIM_Dependency
{
 [Override ("Antecedent"), Description (
  "The physical component that implements the Device.") ]
    CIM_PhysicalElement REF Antecedent;
 [Override ("Dependent"), Description ("The LogicalDevice.") ]
    CIM_LogicalDevice REF Dependent;
};

// ==================================================================
//    Component
// ==================================================================
        [Association, Abstract, Aggregation, Description (
         "CIM_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.") ]
class CIM_Component
{
        [Aggregate, Key, Description (
         "The parent element in the association.") ]
    CIM_ManagedSystemElement REF GroupComponent;
        [Key, Description ("The child element in the association.") ]
    CIM_ManagedSystemElement REF PartComponent;
};

// ==================================================================
//    SystemComponent
// ==================================================================
        [Association, Aggregation, Description (
         "CIM_SystemComponent is a specialization of the CIM_Component "
         "association that establishes 'part of' relationships between "
         "a System and the Managed System Elements of which it is "
   "composed.") ]
class CIM_SystemComponent:CIM_Component
{
        [Override ("GroupComponent"), Aggregate,
         Description ("The parent System in the Association.") ]
    CIM_System REF GroupComponent;
        [Override ("PartComponent"), Description (
         "The child element that is a component of a System.") ]
    CIM_ManagedSystemElement REF PartComponent;
};

// ==================================================================
//    SystemDevice
// ==================================================================
        [Association, Aggregation, Description (
   "LogicalDevices may be aggregated by a System.  This "
   "relationship is made explicit by the SystemDevice "
   "association. ") ]
class CIM_SystemDevice:CIM_SystemComponent
{
```

```
        [Override ("GroupComponent"), Aggregate, Max (1), Min (1),
         Description ("The parent system in the Association.") ]
    CIM_System REF GroupComponent;
        [Override ("PartComponent"), Weak, Description (
         "The LogicalDevice that is a component of a System.") ]
    CIM_LogicalDevice REF PartComponent;
};

// ==================================================================
//    ServiceComponent
// ==================================================================
        [Association, Aggregation, Description (
         "The ServiceComponent aggregation models a set of "
         "subordinate Services that are aggregated together to form "
         "a higher-level service.") ]
class CIM_ServiceComponent : CIM_Component
{
        [Override ("GroupComponent"), Aggregate,
         Description ("The parent Service.") ]
   CIM_Service REF GroupComponent;
        [Override ("PartComponent"),
         Description ("The component Service.") ]
   CIM_Service REF PartComponent;
};

[...]

// ==================================================================
//    StatisticalInformation
// ==================================================================
  [Abstract, Description (
   "CIM_StatisticalInformation is a root class for any arbitrary "
   "collection of statistical data and/or metrics applicable to "
   "one or more ManagedSystemElements.") ]
class CIM_StatisticalInformation
{
  [MaxLen (256), Description (
   "The Name property defines the label by which the statistic "
   "or metric is known. When subclassed, the property can be "
   "overridden to be a Key property. ") ]
    string Name;
  [Description (
   "A textual description of the statistic or metric.") ]
    string Description;
  [MaxLen (64), Description (
   "A short textual description (one-line string) for the "
   "statistic or metric.") ]
    string Caption;
};

// ==================================================================
//    Statistics
// ==================================================================
  [Association, Description (
   "CIM_Statistics is an association that relates ManagedSystem"
   "Elements to the StatisticalGroup(s) that apply to them.") ]
class CIM_Statistics
{
  [Key, Description ("The statistic information/object.") ]
    CIM_StatisticalInformation REF Stats;
  [Key, Description (
   "The ManagedSystemElement for which statistical or metric "
   "data is defined.") ]
    CIM_ManagedSystemElement REF Element;
};

[...]

// ==================================================================
// end of file
// ==================================================================
```

B.2 CIM Application Schema

```
// ==================================================================
// Title      : Application MOF Specification 2.2
// Filename   : CIM_Application22.mof
// Author     : DMTF Application Management Working Group
// Last Change: 06/07/99
// ==================================================================

[...]

// ==================================================================
//    CIM_SoftwareElement
// ==================================================================
        [Description (" The CIM_SoftwareElement class is used to decompose  a "
        "CIM_SoftwareFeature object into a set of individually  manageable or "
        "deployable parts for a particular platform.   A software element's "
        "platform is uniquely identified by its  underlying hardware "
        "architecture and operating system  (for example Sun Solaris on Sun "
        "Sparc or Windows NT on Intel).   As such, to understand the details of "
        "how the functionality  of a particular software feature is provided on "
        "a particular  platform, the CIM_SoftwareElement objects referenced by   "
        "CIM_SoftwareFeatureSoftwareElement associations are organized   in "
        "disjoint sets based on the TargetOperatingSystem property.    A "
        "CIM_SoftwareElement object captures the management  details of a part "
        "or component in one of  four states  characterized by the "
        "SoftwareElementState property.   ") ]
class CIM_SoftwareElement:CIM_LogicalElement
{
        [key, maxlen (256) , override ("Name") , Description (
        "The name used to identify this software element") ]
    string Name ;
        [key, Maxlen (64) , Description (
        "Version should be in the form <Major>.<Minor>.<Revision> or <Major>."
        "<Minor><letter><revision>") , MappingStrings {
        "MIF.DMTF|ComponentID|001.3"} ]
    string Version ;
        [key, Description (
        " The SoftwareElementState is defined in this model to  identify various "
        "states of a software elements life cycle.   - A software element in the "
        "deployable state describes     the details necessary to successful "
        "distribute it and     the details (conditions and actions) required to "
        "create     a software element in the installable state (i.e, the next "
        "state).  - A software element in the installable state describes     "
        "the details necessary to successfully install it and the    details ("
        "conditions and actions required to create a     software element in the "
        "executable state (i.e., the next state).  - A software element in the "
        "executable state describes the     details necessary to successfully  "
        "start it and the details     (conditions and actions required to create "
        "a software element in     the running state (i.e., the next state).  - "
        "A software element in the running state describes the details     "
        "necessary to monitor and operate on a start element.") , Values {
        "Deployable", "Installable", "Executable", "Running"} ]
    uint16 SoftwareElementState ;
        [key, maxlen (256) , Description (
        " This is an identifier for this software element and is designed to be  "
        "used in conjunction with other keys to create a unique representation  "
        "of this SoftwareElement") ]
string SoftwareElementID ;
    [key, MappingStrings {"MIF.DMTF|Software Component Information|002.5"}
    , Description (
    " The Target Operating System property allows the provider to specify "
    "the  operating system environment. The value of this property does not  "
    "ensure binary executable.  Two other pieces of information are needed.  "
    " First, the version of the OS needs to be specified. using the OS  "
    "Version Check. The second piece of information is the architecture the  "
    "OS runs on. This information is capture with the ArchitectureCheck  "
    "class. The combination of these constructs allows the provider to  "
    "clearly identify the level of OS required for a particular software  "
    "element.") , Values {"Unknown", "Other", "MACOS", "ATTUNIX", "DGUX",
    "DECNT", "Digital Unix", "OpenVMS", "HPUX", "AIX", "MVS", "OS400",
    "OS/2", "JavaVM", "MSDOS", "WIN3x", "WIN95", "WIN98", "WINNT", "WINCE",
    "NCR3000", "NetWare", "OSF", "DC/OS", "Reliant UNIX", "SCO UnixWare",
    "SCO OpenServer", "Sequent", "IRIX", "Solaris", "SunOS", "U6000",
    "ASERIES", "TandemNSK", "TandemNT", "BS2000", "LINUX", "Lynx", "XENIX",
    "VM/ESA", "Interactive UNIX", "BSDUNIX", "FreeBSD", "NetBSD", "GNU Hurd"
    , "OS9", "MACH Kernel", "Inferno", "QNX", "EPOC", "IxWorks", "VxWorks",
    "MiNT", "BeOS", "HP MPE", "NextStep", "PalmPilot", "Rhapsody",
    "Windows 2000", "Dedicated","VSE","TPF"} ,
    ModelCorrespondence {"CIM_OperatingSystem.OSType"} ]
uint16 TargetOperatingSystem ;
    [Description (
    " The OtherTargetOS property records the manufacturer and  operating "
    "system type for a software element when  the TargetOperatingSystem "
    "property has a value of  1 (\"Other\").  Therefore, when the "
    "TargetOperating System property has a value of \"Other\", the "
    "OtherTargetOS  property must have a non-null value.  For all other "
    "values  of TargetOperatingSystem, the OtherTargetOS property is to be "
    "NULL. ") , Maxlen (64) , ModelCorrespondence {
    "CIM_OperatingSystem.OtherTypeDescription"} ]
string OtherTargetOS ;
    [Maxlen (256) , MappingStrings {"MIF.DMTF|ComponentID|001.1"} ,
    Description ("Manufacturer of this software element") ]
string Manufacturer ;
    [Maxlen (64) , MappingStrings {
    "MIF.DMTF|Software Component Information|002.4"} , Description (
    "The internal identifier for this compilation of this software element."
    ) ]
string BuildNumber ;
    [Maxlen (64) , MappingStrings {"MIF.DMTF|ComponentID|001.4"} ,
    Description ("The assigned serial number of this software element.") ]
string SerialNumber ;
    [Maxlen (64) , Description (
    "The code set used by this software element. ") ]
string CodeSet ;
    [Maxlen (64) , MappingStrings {
    "MIF.DMTF|Software Component Information|002.7"} , Description (
    " The value of this property is the manufacturer's identifier for this "
    "software element. Often this will be a stock keeping unit (SKU) or a "
    "part number.") ]
string IdentificationCode ;
    [Maxlen (32) , MappingStrings {
    "MIF.DMTF|Software Component Information|002.6"} , Description (
    "The value of this property identifies the language edition of this "
    "software element. The language codes defined in ISO 639 should be used. "
    "Where the software element represents multi-lingual or international "
```

```
        "version of a product, the string multilingual should be used.") ]
    string LanguageEdition ;
};

// ==================================================================
//    CIM_SoftwareFeature
// ==================================================================
        [Description (
        " The CIM_SoftwareFeature class defines a particular function  or "
        "capability of a product or application system.  This  class in intended "
        "to capture the level of granularity that  is meaningful to a consumer "
        "or user of a product rather than  the units that reflect how the "
        "product is built or packaged.  The latter detailed is captured using a "
        "CIM_SoftwareElement  class.   When a software feature can exist on "
        "multiple platforms or  operating systems (for example, a client "
        "component of a  three tiered client/server applications might run on "
        "Solaris,  Windows NT, and Windows 95), a software feature is a "
        "collection  of all the software elements for these different platforms. "
        " In this case, the users of the model must be aware of this situation  "
        "since typically they will be interested in a sub-collection of the  "
        "software elements required for a particular platform.   Software "
        "Features are always defined in the context of a  CIM_Product class "
        "using the CIM_ProductSoftwareFeature  association since features are "
        "delivered through products.  Optionally, software features from one or "
        "more products  can be organized into application systems using the  "
        "CIM_ApplicationSystemSoftwareFeature  association.  ") ]
class CIM_SoftwareFeature:CIM_LogicalElement
{
        [Propagated ("CIM_Product.IdentifyingNumber") , Key, Maxlen (64) ,
        Description (
        "Product identification such as a serial number on software or a die "
        "number on a hardware chip") , MappingStrings {"DMTF|ComponentID|001.4"
        } ]
    string IdentifyingNumber ;
        [Propagated ("CIM_Product.Name") , Key, Maxlen (256) , Description (
        "Commonly used Product name") , MappingStrings {"DMTF|ComponentID|001.2"
        } ]
    string ProductName ;
        [Propagated ("CIM_Product.Vendor") , Key, Maxlen (256) , Description (
        "The name of the Product's supplier. Corresponds to the Vendor property "
        "in the Product object in the DMTF Solution Exchange Standard.") ,
        MappingStrings {"DMTF|ComponentID|001.1"} ]
    string Vendor ;
        [Propagated ("CIM_Product.Version") , Key, Maxlen (64) , Description (
        "Product version information.  Corresponds to the Version property in "
        "the Product object in the DMTF Solution Exchange Standard.") ,
        MappingStrings {"DMTF|ComponentID|001.3"} ]
    string Version ;
        [key, override ("Name") , Description (
        "The Name property defines the label by which the object is known to the "
        "world outside the data processing system. This label is a "
        "human-readable name that uniquely identifies the element in the context "
        "of the element's namespace.") , Maxlen (256) ]
    string Name ;
};

[...]

// ==================================================================
//    SoftwareFeatureSoftwareElements
// ==================================================================
        [Association, Aggregation, Description (
        " The SoftwareFeatureSoftwareElements associations identifies the "
        "software elements that make up a particular software feature.") ]
class CIM_SoftwareFeatureSoftwareElements : CIM_Component
{
        [Min (0) , Max (NULL), Aggregate, Override ("GroupComponent") ]
    CIM_SoftwareFeature REF GroupComponent;
        [Min (0) , Max (NULL), Override ("PartComponent") ]
    CIM_SoftwareElement REF PartComponent;
};

[...]

// ==================================================================
//    SoftwareFeatureServiceImplementation
// ==================================================================
        [Association, Description (
        " An association between a Service and how it is implemented in software."
        "  The cardinality of this association is many-to-many.  A Service may "
        "be  provided by more than one SoftwareFeature, operating in conjunction."
        "   And, any software Feature may provide more than one Service.  When  "
        "multiple SoftwareFeatures are associated with a single Service, it is  "
        "assumed that these elements operate in conjunction to provide the  "
        "Service.  If different implementations of a Service exist, each of  "
        "these implementations would result in individual instantiations of the  "
        "Service object.  These individual instantiations would then have  "
        "associations to the unique implementations. ") ]
class CIM_SoftwareFeatureServiceImplementation : CIM_Dependency
{
        [Min (0) , Max (NULL), Override ("Antecedent") ]
    CIM_SoftwareFeature REF Antecedent;
        [Min (0) , Max (NULL), Override ("Dependent") ]
    CIM_Service REF Dependent;
};

// ==================================================================
//    SoftwareFeatureSAPImplementation
// ==================================================================
        [Association, Description (
        " An association between a ServiceAccessPoint and how it is   "
        "implementedin software.  The cardinality of this association  is "
        "many-to-many.  A SAP may be provided by more than one  SoftwareFeature, "
        "operating in conjunction.  And, any  SoftwareFeature may provide more "
        "than one ServiceAccessPoint. a  When many SoftwareFeatures are "
        "associated with single SAP,  it is assumed that these elements operate "
        "in conjunction to  provide the AccessPoint.  If different "
        "implementations of a  SAP exist, each of these implementations would "
        "result in  individual instantiations of the ServiceAccessPoint object.  "
        "These individual instantiations would then have associations  to the "
        "unique implementations. ") ]
class CIM_SoftwareFeatureSAPImplementation : CIM_Dependency
{
        [Min (0) , Max (NULL), Override ("Antecedent") ]
    CIM_SoftwareFeature REF Antecedent;
        [Min (0) , Max (NULL), Override ("Dependent") ]
```

```
    CIM_ServiceAccessPoint REF Dependent;
};

[...]

// ===================================================================
// end of file
// ===================================================================
```

B.3 CIM Device Schema

```
// ==================================================================
// Title:       Devices MOF specification 2.2
// Filename:    CIM_Device22.mof
// Author:      DMTF System and Devices Working Group
// Last Change: 06/23/1998
// ==================================================================

[...]

// ==================================================================
// DeviceServiceImplementation
// ==================================================================
   [Association,
    Description (
       "An association between a Service and how it is implemented. "
       "The cardinality of this association is many-to-many. "
       "A Service may be provided by more than one Logical"
       "Device, operating in conjunction.  And, any Device may "
       "provide more than one Service.  When multiple Devices are "
       "associated with a single Service, it is assumed that these "
       "elements operate in conjunction to provide the Service.  If "
       "different implementations of a Service exist, each of these "
       "implementations would result in individual instantiations "
       "of the Service object.  These individual instantiations "
       "would then have associations to the unique implementations.")
   ]
class CIM_DeviceServiceImplementation : CIM_Dependency
{
      [Override ("Antecedent"),
       Description ("The LogicalDevice.")
      ]
   CIM_LogicalDevice REF Antecedent;
      [Override ("Dependent"),
       Description ("The Service implemented using the LogicalDevice.")
      ]
   CIM_Service REF Dependent;
};

// ==================================================================
// DeviceSAPImplementation
// ==================================================================
   [Association,
    Description (
       "An association between a ServiceAccessPoint and how "
       "it is implemented. The cardinality of this association "
       "is many-to-many.  A SAP may be provided by more "
       "than one LogicalDevice, operating in conjunction.  And, any "
       "Device may provide more than one ServiceAccessPoint. "
       "When many LogicalDevices are associated with a single "
       "SAP, it is assumed that these elements operate in "
       "conjunction to provide the AccessPoint.  If different "
       "implementations of a SAP exist, each of these implementations "
       "would result in individual instantiations of the "
       "ServiceAccessPoint object.  These individual instantiations "
       "would then have associations to the unique implementations.")
   ]
class CIM_DeviceSAPImplementation : CIM_Dependency
{
      [Override ("Antecedent"),
       Description ("The LogicalDevice.")
      ]
   CIM_LogicalDevice REF Antecedent;
      [Override ("Dependent"),
       Description (
 "The ServiceAccessPoint implemented using the LogicalDevice.")
      ]
   CIM_ServiceAccessPoint REF Dependent;
};

[...]

// ==================================================================
// NetworkAdapter
// ==================================================================
   [Abstract,
    Description (
       "NetworkAdapter is an Abstract class defining general "
       "networking hardware concepts (for example, PermanentAddress or "
       "Speed of operation). NetworkAdapters are Devices with the "
       "ability to support multiple, higher level protocols and provide "
       "the implementation behind the Network Model's ProtocolEndpoint "
       "class. (This information is conveyed using the DeviceSAP"
       "Implementation association, defined in the Core Model.) "
       "NetworkAdapters and their Endpoints represent the potential for "
       "connectivity among peers.\n"
       "The 'potential for connectivity' is very different than the "
       "master-slave/controller-controlled by relationships of CIM_"
       "Controller. Sometimes, however, a single Device is both a kind "
       "of NetworkAdapter and a Controller - for example, when a Fibre"
       "ChannelAdapater is operating as a ComputerSystem's SCSIController. "
       "In this case, there are aspects of the Device that are network "
       "oriented and others that are Controller oriented - and, both the "
       "Controller and Adapter classes should be instantiated. A Device"
       "Identity relationship would also be created to tie together these "
       "differing aspects/abstractions of the Device.")
   ]
class CIM_NetworkAdapter : CIM_LogicalDevice
{
      [MaxLen (64), Description (
  "PermanentAddress defines the network address hardcoded into "
  "an adapter.  This 'hardcoded' address may be changed via "
  "firmware upgrade or software configuration. If so, this field "
  "should be updated when the change is made.  PermanentAddress "
  "should be left blank if no 'hardcoded' address exists for the "
  "NetworkAdapter."),
       MappingStrings {"MIF.DMTF|Network Adapter 802 Port|001.2"}
]
    string PermanentAddress;
[MaxLen (64),
 Description (
  "An array of strings indicating the network addresses for an "
  "adapter."),
 ArrayType ("Indexed"),
 MappingStrings {"MIF.DMTF|Network Adapter 802 Port|001.3"}
```

```
      ]
    string NetworkAddresses[];
[Description (
  "An estimate of the current bandwidth in Bits per Second. "
  "For Adapters which vary in bandwidth or for those where "
  "no accurate estimation can be made, this property should "
  "contain the nominal bandwidth."),
 Units ("Bits per Second"),
 MappingStrings {"MIB.IETF|RFC1213-MIB.ifSpeed",
       "MIF.DMTF|Network Adapter 802 Port|001.5"}
]
    uint64 Speed;
[Description (
  "The maximum speed, in Bits per Second, for the Network"
  "Adapter."),
 Units ("Bits per Second")
]
    uint64 MaxSpeed;
      [Description (
        "Boolean indicating that the Adapter is operating in "
        "full duplex mode.")
      ]
   boolean FullDuplex;
[Description (
  "A boolean indicating whether the NetworkAdapter is capable "
  "of automatically determining the speed or other communications "
        "characteristics of the attached network media.")
]
    boolean AutoSense;
[Description (
  "The total number of octets transmitted, including framing "
  "characters."),
 Mappingstrings {"MIB.IETF|RFC1213-MIB.ifOutOctets",
       "MIF.DMTF|Network Adapter 802 Port|001.7"},
       Counter
      ]
    uint64 OctetsTransmitted;
[Description (
  "The total number of octets received, including framing "
  "characters."),
 Mappingstrings {"MIB.IETF|RFC1213-MIB.ifInOctets",
       "MIF.DMTF|Network Adapter 802 Port|001.9"},
       Counter
      ]
    uint64 OctetsReceived;
};

[...]

// ==================================================================
// EthernetAdapter
// ==================================================================
   [Description ("Capabilities and management of an EthernetAdapter.")]
class CIM_EthernetAdapter : CIM_NetworkAdapter
{
[Override ("NetworkAddresses"),
 Description (
  "Ethernet/802.3 MAC addresses formatted as twelve hexadecimal "
  "digits (e.g. \"010203040506\"), with each pair representing "
  "one of the six octets of the MAC address in \"canonical\" bit "
  "order.  (Thus, the Group address bit is found in the low "
  "order bit of the first character of the string.)"),
 ArrayType ("Indexed")
]
    string NetworkAddresses[];
[Description (
  "The maximum size of the INFO (non-MAC) field that will be "
  "received or transmitted."),
 Mappingstrings {"MIB.IETF|BRIDGE-MIB.dot1dTpPortMaxInfo"}
]
    uint32 MaxDataSize;
      [Description (
        "Capabilities of the EthernetAdapter. For example, the "
        "Device may support AlertOnLan, WakeOnLan, Load Balancing "
        "and/or FailOver. If failover or load balancing "
        "capabilities are listed, a SpareGroup (failover) or "
        "ExtraCapacityGroup (load balancing) should also be defined "
        "to completely describe the capability."),
       ArrayType ("Indexed"),
       Values {"Unknown", "Other", "AlertOnLan", "WakeOnLan",
             "FailOver", "LoadBalancing"},
       ModelCorrespondence {
             "CIM_EthernetAdapter.CapabilityDescriptions"}
      ]
   uint16 Capabilities[];
      [Description (
        "An array of free-form strings providing more detailed "
        "explanations for any of the EthernetAdapter features "
        "indicated in the Capabilities array. Note, each entry of "
        "this array is related to the entry in the Capabilities "
        "array that is located at the same index."),
       ArrayType ("Indexed"),
       ModelCorrespondence {"CIM_EthernetAdapter.Capabilities"}
      ]
   string CapabilityDescriptions[];
      [Description (
        "Specifies which capabilities are enabled from the list "
        "of all supported ones, defined in the Capabilities array."),
       Values {"Unknown", "Other", "AlertOnLan", "WakeOnLan",
             "FailOver", "LoadBalancing"},
       ModelCorrespondence {"CIM_EthernetAdapter.Capabilities"}
      ]
   uint16 EnabledCapabilities[];
      [Description (
        "The number of times there was an invalid data symbol when "
        "a valid carrier was present. The count is incremented at "
        "most once per carrier event, even if multiple symbol "
        "errors occur during the carrier event."),
       MappingStrings {"MIB.IETF|EtherLike-MIB.dot3StatsSymbolErrors"},
       Counter
      ]
   uint32 SymbolErrors;
      [Description ("The total number of packets transmitted."),
       MappingStrings {"MIF.DMTF|Network Adapter 802 Port|001.6"},
       Counter
      ]
```

```
   uint64 TotalPacketsTransmitted;
      [Description ("The total number of packets received."),
       MappingStrings {"MIF.DMTF|Network Adapter 802 Port|001.8"},
       Counter
      ]
   uint64 TotalPacketsReceived;
[Description (
  "A count of frames received on a particular interface "
  "that are not an integral number of octets in length and do "
  "not pass the FCS check. The count represented by an "
  "instance of this object is incremented when the alignment"
  "Error status is returned by the MAC layer to the LLC (or "
  "other MAC user). Received frames for which multiple error "
  "conditions obtain are, according to the conventions of IEEE "
  "802.3 Layer Management, counted exclusively according to "
  "the error status presented to the LLC."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsAlignmentErrors"},
       Counter
      ]
    uint32 AlignmentErrors;
[Description (
  "A count of frames received on a particular interface "
  "that are an integral number of octets in length but do "
  "not pass the FCS check. The count represented by an "
  "instance of this object is incremented when the frame"
  "CheckError status is returned by the MAC layer to the "
  "LLC (or other MAC user). Received frames for which "
  "multiple error conditions obtain are, according to the "
  "conventions of IEEE 802.3 Layer Management, counted "
  "exclusively according to the error status presented to "
  "the LLC."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsFCSErrors"},
       Counter
      ]
    uint32 FCSErrors;
[Description (
  "A count of successfully transmitted frames on a particular "
  "interface for which transmission is inhibited by exactly "
  "one collision. A frame that is counted by an instance of "
  "this object is not counted by the corresponding instance "
  "of the MultipleCollisionFrames property."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsSingleCollisionFrames"},
       Counter
      ]
    uint32 SingleCollisionFrames;
[Description (
  "A count of successfully transmitted frames on a particular "
  "interface for which transmission is inhibited by more than "
  "one collision. A frame that is counted by an instance of "
  "this object is not counted by the corresponding instance "
  "of the SingleCollisionFrames property."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsMultipleCollisionFrames"},
       Counter
      ]
   uint32 MultipleCollisionFrames;
[Description (
  "A count of times that the SQE TEST ERROR message is "
  "generated by the PLS sublayer for a particular interface. "
  "The SQE TEST ERROR message is defined in section "
  "7.2.2.2.4 of ANSI/IEEE 802.3-1985 and its generation is "
  "described in section 7.2.4.6 of the same document."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsSQETestErrors"},
       Counter
      ]
   uint32 SQETestErrors;
[Description (
  "A count of frames for which the first transmission "
  "attempt on a particular interface is delayed because the "
  "medium is busy. The count represented by an instance of "
  "this object does not include frames involved in collisions."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsDeferredTransmissions"},
       Counter
      ]
   uint32 DeferredTransmissions;
[Description (
  "The number of times that a collision is detected on a "
  "particular interface later than 512 bit-times into the "
  "transmission of a packet. Five hundred and twelve bit-"
  "times corresponds to 51.2 microseconds on a 10 Mbit/s "
  "system. A (late) collision included in a count "
  "represented by an instance of this object is also "
  "considered as a (generic) collision for purposes of "
  "other collision-related statistics."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsLateCollisions"},
       Counter
      ]
   uint32 LateCollisions;
[Description (
  "A count of frames for which transmission on a particular "
  "interface fails due to excessive collisions."),
 Mappingstrings {
       "MIB.IETF|EtherLike-MIB.dot3StatsExcessiveCollisions"},
       Counter
      ]
   uint32 ExcessiveCollisions;
[Description (
  "A count of frames for which transmission on a particular "
  "interface fails due to an internal MAC sublayer transmit "
  "error. A frame is only counted by an instance of this "
  "object if it is not counted by the corresponding instance "
  "of either the LateCollisions property, the Excessive"
  "Collisions property, or the CarrierSenseErrors property. "
  "The precise meaning of the count represented by an instance "
  "of this object is implementation-specific.  In particular, "
  "an instance of this object may represent a count of "
  "transmission errors on a particular interface that are "
  "not otherwise counted."),
 Mappingstrings {
  "MIB.IETF|EtherLike-MIB.dot3StatsInternalMacTransmitErrors"},
```

```
      Counter
      ]
    uint32 InternalMACTransmitErrors;
[Description (
  "A count of frames for which reception on a particular "
  "interface fails due to an internal MAC sublayer receive "
  "error. A frame is only counted by an instance of this "
  "object if it is not counted by the corresponding instance "
  "of either the FrameTooLongs property, the AlignmentErrors "
  "property, or the FCSErrors property. The precise meaning "
  "of the count represented by an instance of this object is "
  "implementation-specific.  In particular, an instance of "
  "this object may represent a count of receive errors on a "
  "particular interface that are not otherwise counted."),
 Mappingstrings {
      "MIB.IETF|EtherLike-MIB.dot3StatsInternalMacReceiveErrors"},
      Counter
      ]
    uint32 InternalMACReceiveErrors;
[Description (
  "The number of times that the carrier sense condition was "
  "lost or never asserted when attempting to transmit a frame "
  "on a particular interface. The count represented by an "
  "instance of this object is incremented at most once per "
  "transmission attempt, even if the carrier sense condition "
  "fluctuates during a transmission attempt."),
 Mappingstrings {
      "MIB.IETF|EtherLike-MIB.dot3StatsCarrierSenseErrors"},
      Counter
      ]
    uint32 CarrierSenseErrors;
[Description (
  "A count of frames received on a particular interface that "
  "exceed the maximum permitted frame size. The count "
  "represented by an instance of this object is incremented "
  "when the FrameTooLong status is returned by the MAC layer "
  "to the LLC (or other MAC user). Received frames for which "
  "multiple error conditions obtain are, according to the "
  "conventions of IEEE 802.3 Layer Management, counted "
  "exclusively according to the error status presented to "
  "the LLC."),
 Mappingstrings {
      "MIB.IETF|EtherLike-MIB.dot3StatsFrameTooLongs"},
      Counter
      ]
    uint32 FrameTooLongs;
};

// ==================================================================
// TokenRingAdapter
// ==================================================================
   [Description ("Capabilities and management of a TokenRingAdapter.")]
class CIM_TokenRingAdapter : CIM_NetworkAdapter
{
[Override ("NetworkAddresses"),
 Description (
  "Token Ring/802.5 MAC addresses formatted as twelve "
  "hexadecimal digits (e.g. \"010203040506\"), with each pair "
  "representing one of the six octets of the MAC address in "
  "\"canonical\" bit order.  (Thus, the Group address bit is "
  "found in the low order bit of the first character of the "
  "string.)"),
 ArrayType ("Indexed")
]
    string NetworkAddresses[];
[Description (
  "The maximum size of the INFO (non-MAC) field that will be "
  "received or transmitted."),
 Mappingstrings {"MIB.IETF|BRIDGE-MIB.dot1dTpPortMaxInfo"}
]
    uint32 MaxDataSize;
      [Description (
        "Capabilities of the TokenRingAdapter. For example, the Device "
        "may support AlertOnLan, WakeOnLan, Load Balancing and/or "
        "Failover. If failover or load balancing capabilities are "
        "listed, a SpareGroup (failover) or ExtraCapacityGroup (load "
        "balancing) should also be defined to completely describe the "
        "capability."),
       ArrayType ("Indexed"),
       Values {"Unknown", "Other", "AlertOnLan", "WakeOnLan",
             "FailOver", "LoadBalancing"},
       ModelCorrespondence {
             "CIM_TokenRingAdapter.CapabilityDescriptions"}
      ]
   uint16 Capabilities[];
      [Description (
        "An array of free-form strings providing more detailed "
        "explanations for any of the TokenRingAdapter features "
        "indicated in the Capabilities array. Note, each entry of "
        "this array is related to the entry in the Capabilities array "
        "that is located at the same index."),
       ArrayType ("Indexed"),
       ModelCorrespondence {"CIM_TokenRingAdapter.Capabilities"}
      ]
   string CapabilityDescriptions[];
      [Description (
        "Specifies which of the capabilities from the \"Capabilities\" "
        "property are currently enabled."),
       Values {"Unknown", "Other", "AlertOnLan", "WakeOnLan",
             "FailOver", "LoadBalancing"},
       ModelCorrespondence {"CIM_TokenRingAdapter.Capabilities"}
      ]
   uint16 EnabledCapabilities[];
      [Description (
        "The current status which can be used to diagnose fluctuating "
        "problems that can occur on token rings, after a station has "
        "successfully been added to the ring. Before an open is completed, "
        "this object contains the value indicating \"no status\" (131072). "
        "(The RingState and RingOpenStatus properties are also provided "
        "for debugging problems when the station can not even enter the "
        "ring.) The property's value is a sum of values, one for each "
        "currently applicable condition. The following values are defined "
        "for various conditions:\n"
        "0 = No Problems Detected, 32 = Ring Recovery, "
        "64 = Single Station, 256 = Remove Received, 512 = Reserved, "
        "1024 = Auto-Removal Error, 2048 = Lobe Wire Fault, "
```

```
      "4096 = Transmit Beacon, 8192 = Soft Error, 16384 = Hard Error, "
      "32768 = Signal Loss, 131072 = No Status, Open Not Completed."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5RingStatus"}
    ]
 uint32 RingStatus;
    [Description (
      "The current Device state with respect to entering or leaving "
      "the ring." ),
     Values {"Opened", "Closed", "Opening", "Closing", "Open Failure",
           "Ring Failure"},
     MappingStrings {"MIB.IETF|IEEE 802.5 Token Ring MIB.dot5RingState"}
    ]
 uint16 RingState;
    [Description (
      "This property indicates the success, or the reason for failure, "
      "of the station's most recent attempt to enter the ring." ),
     Values {"No Open Attempted", "Bad Parameter", "Lobe Failed",
           "Signal Loss", "Insertion Timeout", "Ring Failed", "Beaconing",
           "Duplicate MAC", "Request Failed", "Remove Received",
           "Last Open Successful"},
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5RingOpenStatus"}
    ]
 uint16 RingOpenStatus;
    [Description ("The ring's bandwidth."),
     Values {"Unknown", "Other", "One Megabit", "Four Megabit",
           "Sixteen Megabit"},
     MappingStrings {"MIB.IETF|IEEE 802.5 Token Ring MIB.dot5RingSpeed"}
    ]
 uint16 RingSpeed;
    [Description (
      "This counter is incremented when a station detects the absence "
      "of transitions for five half-bit timers (burst-five errors)."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsBurstErrors"},
     Counter
    ]
 uint32 BurstErrors;
    [Description (
      "This counter is incremented when a station receives an AMP or "
      "SMP frame in which A is equal to C is equal to 0, and then "
      "receives another SMP frame with A equal to C equal to 0 without "
      "first receiving an AMP frame. It denotes a station that cannot "
      "set the AC bits properly."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsACErrors"},
     Counter
    ]
 uint32 ACErrors;
    [Description (
      "This counter is incremented when a station transmits an abort "
      "delimiter while transmitting data."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsAbortTransErrors"},
     Counter
    ]
 uint32 AbortTransErrors;
    [Description (
      "This counter is incremented when a station recognizes an "
      "internal error."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsInternalErrors"},
     Counter
    ]
 uint32 InternalErrors;
    [Description (
      "This counter is incremented when a station is transmitting "
      "and its TRR timer expires. This denotes a condition where a "
      "transmitting station in strip mode does not receive the "
      "trailer of the frame before the TRR timer goes off."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsLostFrameErrors"},
     Counter
    ]
 uint32 LostFrameErrors;
    [Description (
      "This counter is incremented when a station recognizes a frame "
      "addressed to its specific address, but has no available buffer "
      "space - indicating that the station is congested."),
    MappingStrings {
     "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsReceiveCongestions"},
     Counter
    ]
 uint32 ReceiveCongestions;
    [Description (
      "This counter is incremented when a station recognizes a frame "
      "addressed to its specific address and detects that the FS field "
      "A bits are set to 1 indicating a possible line hit or duplicate "
      "address."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsFrameCopiedErrors"},
     Counter
    ]
 uint32 FrameCopiedErrors;
    [Description (
      "This counter is incremented when a station acting as the active "
      "monitor recognizes an error condition that needs a token "
      "transmitted."),
     MappingStrings {
     "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsTokenErrors"},
     Counter
    ]
 uint32 TokenErrors;
    [Description (
      "The number of Soft Errors that the Device has detected. It "
      "directly corresponds to the number of Report Error MAC frames "
      "that this Device has transmitted. Soft Errors are those which "
      "are recoverable by the MAC layer protocols."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsSoftErrors"},
     Counter
    ]
 uint32 SoftErrors;
    [Description (
      "The number of times this Device has detected an immediately "
```

```
      "recoverable fatal error. It denotes the number of times this "
      "Device is either transmitting or receiving beacon MAC frames. "),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsHardErrors"},
     Counter
    ]
 uint32 HardErrors;
    [Description (
      "The number of times this Device has detected the loss of "
      "signal condition from the ring."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsSignalLoss"},
     Counter
    ]
 uint32 SignalLossCount;
    [Description (
      "The number of times this Device has transmitted a beacon "
      "frame."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsTransmitBeacons"},
     Counter
    ]
 uint32 TransmittedBeacons;
    [Description (
      "The number of Claim Token MAC frames received or transmitted "
      "after the Device has received a Ring Purge MAC frame. This "
      "counter signifies the number of times the ring has been purged "
      "and is being recovered back into a normal operating state."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsRecoverys"},
     Counter
    ]
 uint32 Recoverys;
    [Description (
      "The number of times the Device has detected an open or short "
      "circuit in the lobe data path. The adapter will be closed and "
      "RingState will signify this condition."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsLobeWires"},
     Counter
    ]
 uint32 LobeWires;
    [Description (
      "The number of times the Device has received a Remove Ring "
      "Station MAC frame request. When this frame is received, the "
      "Device will enter the close state and RingState will "
      "signify this condition."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsRemoves"},
     Counter
    ]
 uint32 Removes;
    [Description (
      "The number of times the Device has sensed that it is the "
      "only station on the ring. This will happen if the Device "
      "is the first one up on a ring, or if there is a hardware "
      "problem."),
    MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsSingles"},
     Counter
    ]
 uint32 Singles;
    [Description (
      "The number of times the Device has detected that the "
      "frequency of the incoming signal differs from the expected "
      "frequency by more than that specified by the IEEE 802.5 "
      "standard."),
     MappingStrings {
      "MIB.IETF|IEEE 802.5 Token Ring MIB.dot5StatsFreqErrors"},
     Counter
    ]
 uint32 FrequencyErrors;
};

[...]

// ==================================================================
// end of file
// ==================================================================
```

B.4 CIM Network Schema

```
// ==================================================================
// Title:       CIM Network Model MOF Specification 2.2
//              (BGP, Multi-Protocol Bridge, and VLAN sub-models.)
// Filename:    CIM_Network22.mof
// Editor:      John Strassner
// Authors:     John Strassner, Al Grimstad, and Thomas McNeill,
//              with valuable input from other Networks WG team
//              members, including Andrea Westerinen, Lee Rafalow,
//              Marc Lavine, Walter Weiss, and Bernie Volz.
//
//              The information model represented here is based on
//              the following RFCs: BGP: 1657, 1700, 1771 , 1965 , 1966,
//              and 1997; Bridge: 1493 and 1525; VLAN includes
//              compliance with the 802.1q specification. Also, MAC
//              address formats are required to be in canonical form,
//              as stated in RFC 2469.
//
// Last Change: 06/10/1999
// ==================================================================

[...]

// ==================================================================
//     AdminDomain
// ==================================================================
  [Abstract, Description (
        "This is a special grouping of ManagedSystemElements that "
        "are all administered by the same user or group of users. "
        "It serves as an aggregation point to associate one or more "
        "of the following elements: network devices, such as "
        "routers and switches, servers, and other resources that "
        "can be accessed by end systems. This grouping of devices "
        "plays an essential role in ensuring that the same "
        "administrative POLICY is applied to all of the devices "
        "in the grouping.") ]

class CIM_AdminDomain : CIM_System
{
        [Override ("NameFormat"), Description (
     "The System class and its subclasses provide the scope for "
        "numerous types of managed objects. As such, these classes "
        "must have the ability to create unique keys. This attribute "
        "is used by the System class and its subclasses to define a "
        "unique Name, independent of the specific discovery protocol "
        "used. Use of the heuristic is optional, but recommended."
        ""
        "AdminDomain is used in the Networks Model to group "
        "together various network resources that must be "
        "administered the same way, perhaps using the same "
        "policies. Viewed in this light, its principal subclass is "
        "AutonomousSystem."
        ""
   "The NameFormat property identifies how the Name of the "
        "AdminDomain is generated, using the heuristic specified "
   "in the CIM V2 System Model spec. It assumes that the "
        "documented rules are traversed in order, to determine and "
         "assign a Name. The NameFormat Values list defines the "
         "precedence order for assigning the Name of the "
         "AdminDomain."),
         ValueMap {"Other", "AS", "NAP", "NOC", "POP", "RNP", "IP", "IPX",
          "SNA", "Dial", "WAN", "LAN", "ISDN", "Frame Relay", "ATM",
          "E.164" },
         Values {"Other", "Autonomous System", "Network Access Provider",
          "Network Operations Center", "Point of Presence", "Regional Network
           Provider", "IP", "IPX", "SNA", "Dial", "WAN", "LAN", "ISDN", "Frame
           Relay", "ATM", "E.164"} ]
    string NameFormat;
};


// ==================================================================
//     NetworkService
// ==================================================================
        [Abstract, Description (
         "This is an abstract base class, derived from the CIM_Service class. "
         "It serves as the root of the network service hierarchy. Network "
         "services represent generic functions that are available from the "
         "network that configure and/or modify the traffic being sent. For "
         "example, FTP is not a network service, as it simply passes data "
         "unchanged from source to destination. On the other hand, services "
         "that provide quality of service (e.g., DiffServ) and security "
         "(e.g., IPSec) do affect the traffic stream. Quality of service, "
         "IPSec, and other services are subclasses of this class. This class "
         "hierarchy enables developers to match services to users, groups, "
         "and other objects in the network.") ]

class CIM_NetworkService : CIM_Service
{
        [Description (
         "This is a free-form array of strings that provide descriptive "
         "words and phrases that can be used in queries to help locate and "
         "identify instances of this service.") ]
    string Keywords [ ];
        [Description (
         "This is a URL that provides the protocol, network location, and "
         "other service-specific information required in order to access the "
         "service. This should be implemented as a LabeledURI, with syntax "
         "DirectoryString and a matching rule of CaseExactMatch, for "
         "directory implementors.") ]
    string ServiceURL;
        [Description (
         "This is a free-form array of strings that specify any specific "
         "pre-conditions that must be met in order for this service to start "
         "correctly. It is expected that subclasses will refine the inherited "
         "StartService() and StopService() methods to suit their own "
         "application-specific needs. This property is used to specify "
         "application-specific conditions needed by the refined StartService "
         "and StopService methods.") ]
    string StartupConditions [ ];
         [Description (
         "This is a free-form array of strings that specify any specific "
         "parameters that must be supplied to the StartService() method in "
         "order for this service to start correctly. It is expected that "
         "subclasses will refine the inherited StartService() and "
```

```
        "StopService() methods to suit their own application-specific "
        "needs. This property is used to specify application-specific "
        "parameters needed by the refined StartService and StopService "
        "methods.") ]
    string StartupParameters [ ];
};

// ==================================================================
//     NetworkServicesInAdminDomain
// ==================================================================
        [Association, Description (
        "This is a specialization of the CIM_HostedService "
        "association that establishes dependency relationships "
        "between an administrative domain and the NetworkServices "
        "that it hosts. This also has the semantics that this set "
        "of services are commonly administered.") ]

class CIM_NetworkServicesInAdminDomain : CIM_HostedService
{
        [Override ("Antecedent"), Min(1), Max(1), Description (
        "The AdminDomain that aggregates the various "
        "network services and administers them.") ]
    CIM_AdminDomain REF Antecedent;
        [Override ("Dependent"), Description (
        "The network service that is hosted in the administrative "
        "domain.") ]
    CIM_NetworkService REF Dependent;
};

// ==================================================================
//    ForwardingService
// ==================================================================
        [Description (
        "This class represents the forwarding of network traffic by "
        "receiving data from one or more ProtocolEndpoints and "
        "sending that data via other ProtocolEndpoints. "
        ""
        "This class is different than the RouteCalculationService "
        "class in that it represents a way to forward traffic "
        "independent of calculating routing information (e.g., using "
        "static routing). Generally, the endpoints among which a "
        "ForwardingService routes data are at the same protocol "
        "layer and are usually of similar types or of the same type. "
        "Examples of this kind of service include routing, bridging, "
        "and repeating services. Instances of this class will have "
        "ForwardsAmong associations to the ProtocolEndpoint instances "
        "among which forwarding is occurring.") ]

class CIM_ForwardingService : CIM_NetworkService
{
        [Description (
        "This defines the type of protocol that is being forwarded. "),
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8",
                   "9", "10", "11", "12" },
        Values { "Unknown", "Other", "IPv4", "IPv6", "IPv4/IPv6",
                 "IPX", "AppleTalk", "DECnet", "SNA", "CONP",
                 "CLNP", "VINES", "XNS" } ]
    uint16 ProtocolType;
        [MaxLen(32), Description (
        "This defines the type of protocol that is being forwarded "
        "when the value of the ProtocolType attribute is 1 (e.g., "
        "'Other'). This provides for future extensibility."),
        ModelCorrespondence { "CIM_ForwardingService.ProtocolType" } ]
    string OtherProtocolType;
};

[...]

// ==================================================================
//     LogicalNetwork
// ==================================================================
        [Description (
        "A LogicalNetwork groups together a set of ProtocolEndpoints of a "
        "given type which are able to communicate with each other "
        "directly. It is used for describing the characteristics of the "
        "grouping and/or its associated medium. A LogicalNetwork "
        "represents the ability to send and/or receive data over a "
        "network.") ]

class CIM_LogicalNetwork : CIM_CollectionOfMSEs
{
        [Propagated ("CIM_System.CreationClassName"), Key,
   MaxLen (256), Description (
        "The scoping System's CreationClassName.") ]
    string SystemCreationClassName;
        [Propagated ("CIM_System.Name"), Key, MaxLen (256),
   Description ("The scoping System's Name.") ]
    string SystemName;
        [Key, MaxLen (256), Description (
        "CreationClassName indicates 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.") ]
    string CreationClassName;
        [Key, MaxLen (256), Description (
        "The Name property defines the label by which the object is "
   "known.") ]
    string Name;
        [MaxLen(64), Description (
        "Type is an enumeration that provides additional "
        "information that can be used to help categorize and "
        "classify different instances of this class . "
        ""
        "Subclasses should ensure that they are of the "
        "appropriate type defined in the Type enumeration. "
        "That is, the IPSubnet subclass should define its "
        "property as either IPv4 or IPv6, as opposed to, "
        "for example, ATM."),
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
                   "10", "11", "12", "13", "14", "15", "16" },
        Values { "Unknown", "Other", "IPv4", "IPv6",
                 "IPX", "AppleTalk", "DECnet", "SNA", "CONP",
                 "CLNP", "VINES", "XNS", "ATM", "Frame Relay",
                 "Ethernet", "TokenRing", "FDDI" },
  ModelCorrespondence {"CIM_LogicalNetwork.OtherTypeDescription"} ]
```

```
   string Type;
        [MaxLen(64), Description (
         "A string describing the type of protocol that is being "
         "run by this LogicalNetwork when the value of the Type "
         "property of the Collection class (or any of its subclasses is "
         "set to 1 (e.g., 'Other'). The format of the string inserted "
         "in this property should be similar in format to the values "
         "defined for the Type property. This property should be set to "
   "NULL when the Type property is any value other than 1."),
 ModelCorrespondence {"CIM_LogicalNetwork.Type"} ]
   string OtherTypeDescription;
};

// ==================================================================
//     LogicalNetworkService
// ==================================================================
        [Association, Description (
         "A LogicalNetworkService represents network services that "
         "either originate and/or terminate in a LogicalNetwork. This "
         "enables management applications to more easily identify "
         "services that are provided by particular network devices.") ]

class CIM_LogicalNetworkService
{
        [Key, Description (
         "The LogicalNetwork that contains the network service.") ]
    CIM_LogicalNetwork REF Network;
        [Key, Description (
   "The NetworkService that is running in the LogicalNetwork. ") ]
    CIM_NetworkService REF NetworkService;
        [Description (
   "An enumeration that explicitly defines this network service "
         "as originating, terminating, or residing in this "
         "LogicalNetwork. "),
        ValueMap { "0", "1", "2", "3" },
        Values {"Unknown", "Originates In", "Terminates In",
         "Resides In" } ]
    uint16 ContainmentType;
};

// ==================================================================
//     IPSubnet
// ==================================================================
        [Description (
         "An IPSubnet represents a group of related IPProtocolEndpoints "
         "that can communicate with each other directly using IP. It is "
         "used for describing the characteristics of the grouping.") ]

class CIM_IPSubnet : CIM_LogicalNetwork
{
        [Description (
         "The IP address of the entire subnet, formatted according "
         "to the appropriate convention as defined in the AddressType "
         "property of this class. "
         ""
         "Note that CIDR format is not supported for this release. "
         "This is because, without more formal methods like constraints, "
         "it is possible to create different instances of the same "
         "object that have different naming formats."
         ""
         "IPv4-compatible addresses should be used instead of IPv6 "
         "addresses (see RFC 2373, section 2.5.4). In order to have a "
         "consistent format for IPv4 addresses in a mixed IPv4/v6 "
         "environment, all IPv4 addresses and both IPv4-compatible IPv6 "
         "addresses and IPv4-mapped IPv6 addresses, per RFC 2373, "
         "section 2.5.4, should be formatted in standard IPv4 format. "
         "However, this (the 2.2) version of the Network Common "
         "Model will not explicitly support mixed IPv4/IPv6 "
         "environments. This will be added in the 2.3 release.") ]
    string SubnetNumber;
        [Description (
         "The mask for the starting IP address of the IPSubnet, formatted "
         "according to the appropriate convention as defined in the "
         "AddressType property of this class (e.g., '255.255.252.0').") ]
    string SubnetMask;
        [Description (
         "An enumeration that describes the format of the address property. "
   "Addresses that can be formatted in IPv4 format, must be formatted "
         "that way to ensure mixed IPv4/IPv6 support."),
        ValueMap { "0", "1", "2" },
        Values { "Unknown", "IPv4", "IPv6" } ]
    uint16 AddressType;
};

// ==================================================================
//     ProtocolEndpoint
// ==================================================================
        [Description (
         "A communication point from which data may be sent or received. "
         "ProtocolEndpoints link router interfaces and switch ports to "
         "LogicalNetworks.") ]

class CIM_ProtocolEndpoint : CIM_ServiceAccessPoint
{
        [Override ("Name"), MaxLen(256), Description (
         "A string which identifies this ProtocolEndpoint with either a "
         "port or an interface on a device. To ensure uniqueness, the "
         "Name property should be prepended or appended with information "
         "from the Type or OtherTypeDescription properties. The method "
         "chosen is described in the NameFormat property of this class.") ]
    string Name;
        [MaxLen (256), Description (
         "NameFormat contains the naming heuristic that is chosen to "
         "ensure that the value of the Name property is unique. For "
         "example, one might choose to prepend the name of the port or "
         "interface with the Type of ProtocolEndpoint that this instance "
         "is (e.g., IPv4)followed by an underscore.") ]
    string NameFormat;
        [MaxLen (64), Description (
         "Type is an enumeration that provides additional information "
         "that can be used to help categorize and classify different "
         "instances of this class."),
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
                   "10", "11", "12", "13", "14", "15", "16" },
        Values { "Unknown", "Other", "IPv4", "IPv6",
                 "IPX", "AppleTalk", "DECnet", "SNA", "CONP",
```

```
                "CLNP", "VINES", "XNS", "ATM", "Frame Relay",
                "Ethernet", "TokenRing", "FDDI" },
  ModelCorrespondence {"CIM_ProtocolEndpoint.OtherTypeDescription"} ]
    string Type;
        [MaxLen(64), Description (
         "A string describing the type of ProtocolEndpoint that this "
         "instance is when the Type property of this class (or any of "
         "its  subclasses) is set to 1 (e.g., 'Other'). The format of "
         "the string inserted in this property should be similar in "
         "format to the values defined for the Type property. This "
         "property should be set to NULL when the Type property is any "
         "value other than 1."),
  ModelCorrespondence {"CIM_ProtocolEndpoint.Type"} ]
   string OtherTypeDescription;
        [Description (
         "An estimate of the current bandwidth in Bits per Second. "
         "For endpoints which vary in bandwidth or for those where "
         "no accurate estimation can be made, this property should "
         "contain the nominal bandwidth."),
        Units ("Bits per Second"),
        MappingStrings {"MIB.IETF|RFC1213-MIB.ifSpeed" } ]
   uint64 Speed;
        [Description (
         "The maximum speed, in Bits per Second, for the "
         "ProtocolEndpoint."),
        Units ("Bits per Second") ]
    uint64 MaxSpeed;
};

// ==================================================================
//    ForwardsAmong
// ==================================================================
        [Association, Description (
         "This association represents the dependency that exists between "
         "the ProtocolEndpoints that are used to forward data and the "
         "ForwardingService that is performing the forwarding of data.") ]

class CIM_ForwardsAmong : CIM_ServiceSAPDependency
{
        [Override ("Antecedent"), Description(
         "The ProtocolEndpoints that are used to forward the data") ]
    CIM_ProtocolEndpoint REF Antecedent;
        [Override ("Dependent"), Description(
         "The service that is forwarding the data") ]
    CIM_ForwardingService REF Dependent;
};

// ==================================================================
//     LANEndpoint
// ==================================================================

        [Description (
         "A communication endpoint which, when its associated interface "
         "device is connected to a LAN, may send and receive data "
         "frames. LANEndpoints link switch ports and host interfaces "
         "to LANs. ") ]

class CIM_LANEndpoint : CIM_ProtocolEndpoint
{
        [Description (
         "A label or identifier for the LAN Segment."),
  ModelCorrespondence {"CIM_LANSegment.LANID"} ]
    string LANID;
        [Description (
         "An indication of the kind of technology used on the LAN."),
        ValueMap { "0", "1", "2", "3", "4" },
        Values { "Unknown", "Other", "Ethernet", "TokenRing", "FDDI" },
        ModelCorrespondence {"CIM_LANSegment.LANType"} ]
    uint16 LANType;
        [Description (
         "A free-form string that describes the type of technology "
         "used on the LAN when the value of the LANType property "
         "is equal to 1 (e.g., 'Other'). This provides built-in "
         "extensibility."),
        ModelCorrespondence {"CIM_LANEndpoint.LANType"} ]
    string OtherLANType;
        [MaxLen(12), Description (
         "The principal unicast address used in communication "
         "with the LANEndpoint. The MAC address is formatted as "
         "twelve hexadecimal digits (e.g., \"010203040506\"), "
         "with each pair representing one of the six octets "
         "of the MAC address in \"canonical\" bit order "
         "according to RFC 2469.") ]
    string MACAddress;
        [Description (
         "Other unicast addresses that may be used to communicate "
         "with the LANEndpoint.") ]
    string AliasAddresses[];
        [Description (
         "Multicast addresses to which the LANEndpoint "
         "listens.") ]
    string GroupAddresses[];
        [Description (
         "The largest information field that may be sent or received "
         "by the LANEndpoint."), Units ("Bits") ]
    uint32 MaxDataSize;
};

// ==================================================================
//     LANSegment
// ==================================================================
        [Description (
         "A collection of LAN Endpoints of a particular type "
         "that are able to intercommunicate directly without "
         "the assistance of bridging or routing services.") ]

class CIM_LANSegment : CIM_LogicalNetwork
{
        [MaxLen(64), Description (
         "A label or identifier for the LAN Segment."),
  ModelCorrespondence {"CIM_LANEndpoint.LANID"} ]
    string LANID;
        [Description (
         "An indication of the kind of technology used on the LAN."),
        ValueMap { "0", "1", "2", "3", "4" },
        Values { "Unknown", "Other", "Ethernet", "TokenRing", "FDDI" },
```

```
        ModelCorrespondence {"CIM_LANEndpoint.LANType"} ]
    uint16 LANType;
        [MaxLen(64), Description (
         "A string describing the type of technology used on the LAN "
         "when the value of the LANType property of this class (or "
         "any of its subclasses) is set to 1 (e.g., 'Other'). The "
         "format of the string inserted in this property should be "
         "similar in format to the values defined for the LANType "
         "property. This property should be set to NULL when the "
         "LANType property is any value other than 1."),
 ModelCorrespondence {"CIM_LANSegment.LANType"} ]
   string OtherLANType;
};

// ==================================================================
//    InLogicalNetwork
// ==================================================================
        [Association, Aggregation, Description (
         "This association defines a ProtocolEndpoint as a member of a "
         "specific LogicalNetwork.") ]

class CIM_InLogicalNetwork : CIM_CollectedMSEs
{
        [Aggregate, Max(1), Override ("Collection"), Description (
         "The LogicalNetwork that groups the ProtocolEndpoint instances.") ]
    CIM_LogicalNetwork REF Collection;
        [Override ("Member"), Description (
         "The child ProtocolEndpoints that are components of the "
         "LogicalNetwork.") ]
    CIM_ProtocolEndpoint REF Member;
};

// ==================================================================
//    InSegment
// ==================================================================
        [Association, Aggregation, Description (
         "Defines a LANEndpoint as a member of a specific LANSegment.") ]

class CIM_InSegment : CIM_InLogicalNetwork
{
        [Aggregate, Max(1), Override ("Collection"), Description (
         "The LANSegment that groups the LANEndpoint instances.") ]
    CIM_LANSegment REF Collection;
        [Override ("Member"), Description (
         "A child LANEndpoint that is a component of the LANSegment.") ]
    CIM_LANEndpoint REF Member;
};

// ==================================================================
//    ActiveConnection
// ==================================================================
        [Association, Description (
         "This association defines a connection that is currently "
         "carrying traffic between two ProtocolEndpoints.") ]

class CIM_ActiveConnection : CIM_SAPSAPDependency
{
        [Override ("Antecedent"), Description (
         "The source ProtocolEndpoint") ]
    CIM_ProtocolEndpoint REF Antecedent;
        [Override ("Dependent"), Description (
         "The destination ProtocolEndpoint") ]
    CIM_ProtocolEndpoint REF Dependent;
        [Description ("The type of traffic that is carried "
         "over this connection"),
        ValueMap { "0", "1", "2", "3", "4", "5" },
        Values { "Unknown", "Other", "Unicast", "Broadcast",
         "Multicast", "Anycast" },
  ModelCorrespondence {
         "CIM_ActiveConnection.OtherTrafficDescription"} ]
    uint16 TrafficType;
        [MaxLen(64), Description (
         "A string describing the type of traffic that is being "
         "carried over this instance when its Type property is set "
         "to 1 (e.g., 'Other'). The format of the string inserted in this "
         "property should be similar in format to the values defined for "
         "the Type property. This property should be set to NULL when the "
         "Type property is any value other than 1."),
  ModelCorrespondence {"CIM_ActiveConnection.Type"} ]
    string OtherTrafficDescription;
        [Description (
         "TRUE means that this connection is uni-directional; FALSE "
         "means that this connection is bi-directional.") ]
    boolean IsUnidirectional;
};

// ==================================================================
//     IPAddressRange
// ==================================================================
        [Abstract, Description (
         "This represents a grouping of specific addresses, and "
         "provides additional semantics for this group if appropriate.") ]

class CIM_IPAddressRange : CIM_CollectionOfMSEs
{
        [Description (
         "The starting IP address of the AddressRange, formatted according "
         "to the appropriate convention as defined in the AddressType "
         "property of this class (e.g., '171.79.6.40').") ]
    string StartAddress;
        [Description (
         "The ending IP address of the AddressRange, formatted according "
         "to the appropriate convention as defined in the AddressType "
         "property of this class (e.g., '171.79.6.40').") ]
    string EndAddress;
        [Description (
         "An enumeration that defines how to format the address and mask"
         "of the address range that defines this IPSubnet)."
         ""
         "Whenever possible, IPv4-compatible addresses should "
         "be used instead of IPv6 addresses (see RFC 2373, "
         "section 2.5.4). In order to have a consistent format "
         "for IPv4 addresses in a mixed IPv4/v6 environment, all "
         "IPv4 addresses and both IPv4-compatible IPv6 addresses "
         "and IPv4-mapped IPv6 addresses, per RFC 2373, section "
         "2.5.4, should be formatted in standard IPv4 format. "
```

```
         "However, this (the 2.2) version of the Network Common "
         "Model will not explicitly support mixed IPv4/IPv6 "
         "environments. This will be added in the 2.3 release."),
        ValueMap { "0", "1", "2" },
        Values { "Unknown", "IPv4", "IPv6" } ]
    uint16 TypeOfAddress;
        [Description (
         "This attribute defines the region that addresses can be "
         "allocated to."),
        ValueMap {"0", "1", "2", "3", "4", "5", "6", "7", "8" },
        Values {"Unknown",
                "Multiregional: 192.0.0.0 to 193.255.255.255",
                "Europe: 194.0.0.0 to 195.255.255.255",
                "Others: 196.0.0.0 to 197.255.255.255",
                "North America: 198.0.0.0 to 199.255.255.255",
                "Central & South America: 200.0.0.0 to 201.255.255.255",
                "Pacific Rim: 202.0.0.0 to 203.255.255.255",
                "Others: 204.0.0.0 to 205.255.255.255",
                "Others: 206.0.0.0 to 207.255.255.255" } ]
    uint16 AllocationRegion;
};

// ==================================================================
//     IPProtocolEndpoint
// ==================================================================
        [Description (
         "A ProtocolEndpoint that is dedicated to running IP.") ]

class CIM_IPProtocolEndpoint : CIM_ProtocolEndpoint
{
        [Description (
         "The IP address that this ProtocolEndpoint represents, formatted "
         "according to the appropriate convention as defined in the "
         "AddressType property of this class (e.g., '171.79.6.40').") ]
    string Address;
        [Description (
         "The mask for the IP address of this ProtocolEndpoint, formatted "
         "according to the appropriate convention as defined in the "
         "AddressType property of this class (e.g., '255.255.252.0').") ]
    string SubnetMask;
        [Description (
         "An enumeration that describes the format of the address "
         "property. Whenever possible, IPv4-compatible addresses should "
         "be used instead of native IPv6 addresses (see RFC 2373, "
         "section 2.5.4). In order to have a consistent format for IPv4 "
         "addresses in a mixed IPv4/v6 environment, all IPv4 addresses "
         "and both IPv4-compatible IPv6 addresses and IPv4-mapped IPv6 "
         "addresses, per RFC 2373, section 2.5.4, should be formatted in "
         "standard IPv4 format. However, this (the 2.2) version of the "
         "Network Common Model will not explicitly support mixed "
         "IPv4/IPv6 environments. This will be added in the "
         "2.3 release."),
        ValueMap { "0", "1", "2" },
        Values { "Unknown", "IPv4", "IPv6" } ]
    uint16 AddressType;
        [Description (
         "It is not possible to tell from the address alone if a given "
         "IPProtocolEndpoint can support IPv4 and IPv6, or just one of "
         "these. This property explicitly defines the support for "
         "different versions of IP that this IPProtocolEndpoint has. "
         ""
         "More implementation experience is needed in order to correctly "
         "model mixed IPv4/IPv6 networks; therefore, this version (2.2) "
         "of the Network Common Model will not support mixed IPv4/IPv6 "
         "environments. This will be looked at more in the 2.3 version."),
        ValueMap { "0", "1", "2" },
        Values { "Unknown", "IPv4 Only", "IPv6 Only" } ]
    uint16 IPVersionSupport;
};

[...]

// ==================================================================
//    IPXProtocolEndpoint
// ==================================================================
        [Description (
         "An IPX communication point from which data may be sent "
         "or received.") ]

class CIM_IPXProtocolEndpoint : CIM_ProtocolEndpoint
{
        [Description (
         "An IPX address formatted as eight hexadecimal digits representing "
         "the network number, followed by a colon, followed by twelve "
         "hexadecimal digits representing the host address "
         "(e.g. \"00112233:010203040506\").") ]
    string Address;
};

// ==================================================================
//    IPXNetwork
// ==================================================================
        [Description ("A network or subnet that uses the IPX protocol") ]

class CIM_IPXNetwork : CIM_LogicalNetwork
{
        [Description (
         "An IPX network number formatted as eight hexadecimal digits "
         " (e.g., \"00112233\").") ]
    string NetworkNumber;
};


// ==================================================================
//     IPRoute
// ==================================================================
        [Abstract, Description (
         "An IPRoute relates a remote IP address (destination) to a "
         "local IP address (source) through which the remote address "
         "may be reached. The local and remote addresses may be either "
         "specific IP endpoints or IP subnets. This class may used to "
         "represent a generic routing table entry.") ]

class CIM_IPRoute : CIM_LogicalElement
{
        [Propagated("CIM_System.CreationClassName"),
```

```
         Key, MaxLen (256), Description (
          "The scoping System's CreationClassName.") ]
    string SystemCreationClassName;
        [Propagated("CIM_System.Name"), Key, MaxLen (256),
         Description (
          "The scoping System's Name.") ]
    string SystemName;
        [Propagated("CIM_Service.CreationClassName"),
         Key, MaxLen (256), Description (
          "The scoping Service's CreationClassName.") ]
    string ServiceCreationClassName;
        [Propagated("CIM_Service.Name"), Key, MaxLen (256),
         Description (
          "The scoping Service's Name.") ]
    string ServiceName;
        [Key, MaxLen (256), Description (
   "CreationClassName indicates 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.") ]
    string CreationClassName;
        [Key, Description (
         "The IP address which serves as the destination of the traffic, "
         "formatted according to the appropriate convention as defined "
         "in the AddressType property of this class.") ]
    string IPDestinationAddress;
        [Key, Description (
         "The mask for the destination IP address, formatted according "
         "to the appropriate convention as defined in the AddressType "
         "property of this class.") ]
    string IPDestinationMask;
        [Key, Description (
         "This contains either the address of the directly connected "
         "interface of the next-hop router or the address of the "
         "interface to which the destination is connected to.") ]
    string NextHop;
        [Key, Description (
         "An enumeration that describes the format of the address property. "
   "Addresses that can be formatted in IPv4 format, must be formatted "
         "that way to ensure mixed IPv4/IPv6 support. "
         ""
         "AddressType is part of the key so that an IPv4 and an IPv6 route "
         "to IP subnets with the same network number but different versions "
         " (v4/v6) can coexist "),
        ValueMap { "0", "1", "2" },
        Values { "Unknown", "IPv4", "IPv6" } ]
    uint16 AddressType;
        [Description (
         "TRUE indicates that this ProtocolEndpoint represents a "
         "static route, and FALSE means that it represents a "
         "dynamically-learned route.") ]
    boolean IsStatic;
};

[...]

// ==================================================================
//    BindsTo
// ==================================================================
        [Association, Description (
         "This association establishes a ProtocolEndpoint on one system "
         "as a requestor of protocol services from another ProtocolEndpoint "
         "on the same system.  This binding establishes a layering of two "
         "protocols, the upper layer represented by the Dependent and the "
         "lower layer represented by the Antecedent.") ]

class CIM_BindsTo : CIM_SAPSAPDependency
{
        [Override ("Antecedent") ]
    CIM_ServiceAccessPoint REF Antecedent;
        [Override ("Dependent") ]
    CIM_ProtocolEndpoint REF Dependent;
};

// ==================================================================
//    BindsToLANEndpoint
// ==================================================================
        [Association, Description (
         "This association makes explicit the dependency of a "
         "ProtocolEndpoint on some lower layer ProtocolEndpoint on "
         "the same system.") ]

class CIM_BindsToLANEndpoint : CIM_BindsTo
{
        [Override ("Antecedent") ]
    CIM_LANEndpoint REF Antecedent;
        [Description (
         "This describes the framing method for the upper layer protocol "
         "endpoint that is bound to a LANEndpoint. Note: \"Raw802.3\" is "
         "only known to be used with the IPX protocol."),
        ValueMap { "0", "1", "2", "3", "4" },
        Values { "Unknown", "Ethernet", "802.2", "SNAP", "Raw802.3" } ]
    uint16 FrameType;
};

[...]

// ==================================================================
//     NetworksInAdminDomain
// ==================================================================
        [Association, Description (
         "This association is used to define the set of LogicalNetworks "
         "that it contains. This has the implied semantics that this "
         "set of LogicalNetworks are all administered by the same "
         "network administrator that administrates the AdminDomain.") ]

class CIM_NetworksInAdminDomain
{
        [Key, Min(1), Max(1), Description (
         "The AdminDomain that hosts the various LogicalNetworks.") ]
    CIM_AdminDomain REF Domain;
        [Key, Weak, Description (
         "The LogicalNetworks that are hosted by the AdminDomain.") ]
    CIM_LogicalNetwork REF Network;
};
```

```
[...]

// ==================================================================
//    ProvidesEndpoint
// ==================================================================
        [Association, Description (
         "A protocol service provides endpoints to give access its "
         "services.") ]

class CIM_ProvidesEndpoint : CIM_ServiceAccessBySAP
{
        [Override ("Antecedent"), Max(1), Description (
         "The NetworkService that is being accessed by the "
         "ProtocolEndpoints.") ]
    CIM_NetworkService REF Antecedent;
        [Override ("Dependent"), Description (
         "The ProtocolEndpoints that provide access to the "
         "NetworkService.") ]
    CIM_ProtocolEndpoint REF Dependent;
};

[...]

// ==================================================================
//    HostedForwardingServices
// ==================================================================
        [Association, Description (
         "This is a specialization of CIM_HostedService, which is an"
         "association between a Service and the System on which the "
         "functionality resides. The ComputerSystem is made up of "
         "different forwarding services, which take the form of internal "
         "and external routing protocols. These are different "
         "NetworkServices, and are weak with respect to their hosting "
         "ComputerSystem.") ]

class CIM_HostedForwardingServices : CIM_HostedService
{
        [Override ("Antecedent"), Max(1), Min(1),
   Description ("The hosting router.") ]
    CIM_ComputerSystem REF Antecedent;
        [Override ("Dependent"), Description (
         "The Service hosted on the System.") ]
    CIM_ForwardingService REF Dependent;
};

[...]

// ===================================================================
//     SwitchService
// ===================================================================
   [Description (
       "Generic switch (bridging) service class. Additional "
       "switching functions are incorporated as subordinate "
       "services related to this class via ServiceComponent "
       "associations.")
   ]
class CIM_SwitchService : CIM_ForwardingService
{
      [MaxLen(12), Description (
          "MAC address used by this switch service when it must be "
          "uniquely identified. When concatenated with a "
          "SpanningTreeService Priority, a unique bridge identifier "
          "results. The MAC address is formatted as twelve "
          "hexadecimal digits (e.g., \"010203040506\"), with each "
          "pair representing one of the six octets of the MAC address "
          "in \"canonical\" bit order according to RFC 2469."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dBaseBridgeAddress"}
      ]
    string BridgeAddress ;
      [Description (
          "The number of switch ports controlled by this "
          "switching service.") ,
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dBaseNumPorts"} ]
    uint16 NumPorts ;
      [Description (
          "Indicates what type of switching service can be performed."),
       ValueMap {"1", "2", "3", "4"},
        Values {"unknown", "transparent-only", "sourceroute-only", "srt"},
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dBaseType"} ]
    uint8 BridgeType ;
};


// ==================================================================
//     SwitchPort
// ==================================================================
   [Description (
       "Switch Port from which frames are received "
       "and out which they are transmitted.")
   ]
class CIM_SwitchPort : CIM_ProtocolEndpoint
{
      [Description (
          "Numeric identifier for a switch port."),
          Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dPort"}
      ]
   uint16 PortNumber;
};

// ==================================================================
//     SwitchesAmong
// ==================================================================
   [Association,
    Description (
       "A SwitchService switches frames between SwitchPorts. "
       "This association makes that relationship explicit.")
   ]
class CIM_SwitchesAmong : CIM_ForwardsAmong
{
      [Override ("Antecedent"),
       Description ("The switch port.")
      ]
   CIM_SwitchPort REF Antecedent;
      [Override ("Dependent"),  Max(1),
       Description ("The switching service.")
```

```
      ]
   CIM_SwitchService REF Dependent;
};

// ===================================================================
//     Switchable
// ===================================================================
   [Association,
    Description (
       "A switch port has a LANEndpoint that is "
       "exposed via this relationship.")
   ]
class CIM_Switchable : CIM_BindsTo
{
   [Override ("Antecedent"), Max(1),
    Description ("The switch port.")
   ]
   CIM_SwitchPort REF Antecedent;
   [Override ("Dependent"), Max(1),
    Description ("The LAN endpoint.")
   ]
   CIM_LANEndpoint REF Dependent;
};

// ===================================================================
//     Transparent Bridging Service
// ===================================================================
   [Description (
       "This service class represents the learning/transparent "
       "bridging aspect of switch service.")
   ]
class CIM_TransparentBridgingService : CIM_ForwardingService
{
      [Description (
          "The timeout period in seconds for aging out "
          "dynamically learned forwarding information. "
          "802.1D-1990 recommends a default of 300 seconds."),
       Units("Seconds"),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dTpAgingTime"}
      ]
   uint32 AgingTime;
      [Description (
          "Filtering Database Identifier. Used by VLAN-aware "
          "switch having more than one filtering database."),
       Mappingstrings {"MIB.IETF|RFC????-MIB.dot1qFdbId"}
      ]
   uint32 FID;
};

// ===================================================================
//     Spanning Tree Service
// ===================================================================
   [Description (
       "This service class represents the capability of a switch "
       "to participate in the distributed construction of a "
       "spanning tree.")
   ]
class CIM_SpanningTreeService : CIM_ForwardingService
{
      [Description (
          "The version of the spanning tree protocol "
          "used by the switch."),
       ValueMap {"1", "2", "3"},
        Values {"unknown", "decLb100", "ieee802d"},
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpProtocolSpecification"}
      ]
   uint16 ProtocolSpec;
      [Description (
          "The format of the priority bits are the first two octets of the "
          "8-octet long Bridge ID. The other (last) 6 octets of the "
          "Bridge ID are given by the value of the Bridge Address. "
          ""
          "A priority that can be assigned to the switch for use in "
          "constructing the spanning tree. The unique identifier for "
          "a switch is constructed by concatenating the MAC address "
          "associated with the switch for spanning tree operations "
          "to the two-byte priority. Choice of the priority value "
          "influences election of the root bridge."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPriority"}
      ]
   uint16 Priority;
      [MaxLen(16), Description (
          "The Bridge ID (Priority + MAC Address) of the root bridge."
          ""
          "The format of the priority bits are the first two octets of the "
          "8-octet long Bridge ID. The other (last) 6 octets of the "
          "Bridge ID are given by the value of the Bridge Address. "
          "Note that the MAC address is formatted as twelve "
          "hexadecimal digits (e.g., \"010203040506\"), with each "
          "pair representing one of the six octets of the MAC address "
          "in \"canonical\" bit order according to RFC 2469. The "
          "priority "),
      Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpDesignatedRoot"}
      ]
   string DesignatedRoot;
      [Description (
          "The cost of the path from the switch to the root."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpRootCost"}
      ]
   uint16 RootCost;
      [Description (
          "The port number of the port having the lowest cost path "
          "to the root bridge."),
      Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpRootPort"}
      ]
   uint16 RootPort;
      [Description (
          "The current value for the maximum age of Spanning Tree "
          "information before discard, as learned from the network."),
      Units ("Hundredths of Seconds"),
      Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpMaxAge"}
      ]
   uint32 MaxAge;
      [Description (
          "The current value of the interval between transmission "
          "of bridge PDUs by the switch on any port for which it is "
```

```
        "the spanning tree root or trying to become so, as learned "
        "from the network."),
    Units ("Hundredths of Seconds"),
    Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpHelloTime"}
    ]
 uint32 HelloTime;
    [Description (
        "The minimum interval between transmission of bridge PDUs "
        "through a given port, as learned from the network."),
    Units ("Hundredths of Seconds"),
    Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpHoldTime"}
    ]
 uint32 HoldTime;
    [Description (
        "The time spent by a port in the listening state before "
        "transitioning to the learning state and in the learning "
        "state before transitioning to the forwarding state, as "
        "learned from the network. This value is also used during "
        "the period of a topology change as the maximum age of "
        "Spanning Tree information before discard. "),
    Units ("Hundredths of Seconds"),
    Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpForwardDelay"}
    ]
 uint32 ForwardDelay;
    [Description (
        "The MaxAge parameter to be used by all switches in the "
        "network if this switch becomes the root."),
    Units ("Hundredths of Seconds"),
    Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpBridgeMaxAge"}
    ]
 uint32 BridgeMaxAge;
    [Description (
        "The HelloTime parameter to be used by all switches in the "
        "network if this switch becomes the root."),
    Units ("Hundredths of Seconds"),
    Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpBridgeHelloTime"}
    ]
 uint32 BridgeHelloTime;
    [Description (
        "The ForwardDelay parameter to be used by all switches in "
"the network if this switch becomes the root."),
    Units ("Hundredths of Seconds"),
    Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpBridgeForwardDelay"}
    ]
 uint32 BridgeForwardDelay;
    [Description (
        "MAC address used by the spanning tree service when it "
        "must be uniquely identified. When concatenated with a "
        "SpanningTreeService Priority, a unique bridge identifier "
        "results. This property is defined for use of switches "
        "supporting multiple spanning tree services. "
        ""
        "Note that the MAC address is formatted as twelve "
        "hexadecimal digits (e.g., \"010203040506\"), with each "
        "pair representing one of the six octets of the MAC address "
        "in \"canonical\" bit order according to RFC 2469.")
    ]
 string SpanningTreeBridgeAddress;
};

// ==================================================================
//     SwitchPortSpanningTree
// ==================================================================
   [Association,
    Description (
       "A switch participating in the spanning tree maintains "
       "spanning-tree specific information about each port "
       "whose forwarding is determined by the spanning tree. "
       "This association represents that information.")
   ]
class CIM_SwitchPortSpanningTree : CIM_ForwardsAmong
{
      [Override ("Antecedent"), Description ("The switch port.")]
   CIM_SwitchPort REF Antecedent;
      [Override ("Dependent"),
       Description ("The spanning tree service.")
      ]
   CIM_SpanningTreeService REF Dependent;
      [Description (
          "The priority assigned to the port. Contained in "
          "the first octet of the two-octet port ID; the other "
          "octet is the port number."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortPriority"}
      ]
   uint8 Priority;
      [Description (
          "The current state of the port as determined by the "
          "spanning tree protocol."),
       ValueMap {"1", "2", "3", "4", "5", "6"},
        Values {"disabled", "blocking", "listening", "learning",
                "forwarding", "broken"},
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortState"}
      ]
   uint16 State;
      [Description (
          "The enabled/disabled status of the port."),
       ValueMap {"1", "2"},
        Values {"enabled", "disabled"},
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortEnable"}
      ]
   uint16 Enable;
      [Description (
          "The contribution of this port to the path cost of "
          "paths towards the spanning tree root which include "
          "this port."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortPathCost"}
      ]
   uint16 PathCost;
      [Description (
          "The bridge identifier of the root bridge for the segment "
          "to which the port is attached, as transmitted by the "
          "designated bridge for the segment."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortDesignatedRoot"}
      ]
   string DesignatedRoot;
      [Description (
```

```
          "The cost of the path to the root offered by the "
          "designated bridge for the segment."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortDesignatedCost"}
      ]
   uint16 DesignatedCost;
      [Description (
          "The bridge identifier of the designated bridge for the "
          "segment to which the port is attached."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortDesignatedBridge"}
      ]
   string DesignatedBridge;
      [Description (
          "The port identifier of the port on the designated bridge "
          "serving the segment to which the port is attached."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStpPortDesignatedPort"}
      ]
   uint16 DesignatedPort;
};

// ===================================================================
//     DynamicForwardingEntry
// ===================================================================
   [Description (
       "A DynamicForwardingEntry represents an entry in the "
       "forwarding (filtering) database associated with the "
       "transparent bridging service.")
   ]
class CIM_DynamicForwardingEntry : CIM_LogicalElement
{
      [Propagated("CIM_System.CreationClassName"),
       Key, MaxLen (256),
       Description ("The scoping System's CreationClassName.")
      ]
   string SystemCreationClassName;
      [Propagated("CIM_System.Name"),
       Key, MaxLen (256),
       Description ("The scoping System's Name.")
      ]
   string SystemName;
      [Propagated("CIM_Service.CreationClassName"),
       Key, MaxLen (256),
       Description ("The scoping Service's CreationClassName.")
      ]
   string ServiceCreationClassName;
      [Propagated("CIM_Service.Name"),
       Key, MaxLen (256),
       Description ("The scoping Service's Name.")
      ]
   string ServiceName;
      [Key, MaxLen (256),
       Description (
  "CreationClassName indicates 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.")
      ]
   string CreationClassName;
      [Key, MaxLen (6), Description (
          "Unicast MAC address for which the transparent bridging "
          "service has forwarding and/or filtering information. "
          ""
          "Note that the MAC address is formatted as twelve "
          "hexadecimal digits (e.g., \"010203040506\"), with each "
          "pair representing one of the six octets of the MAC address "
          "in \"canonical\" bit order according to RFC 2469."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dTpFdbAddress"}
      ]
   string MACAddress;
      [Description (
          "The status of the entry."),
       ValueMap {"1", "2", "3", "4", "5"},
        Values {"other", "invalid", "learned", "self", "mgmt"},
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dTpFdbStatus"}
      ]
   uint16 DynamicStatus;
};

// ==================================================================
//    TransparentBridgingDynamicForwarding
// ==================================================================
   [Association,
    Description (
       "This association links an instance of the transparent "
       "bridging service to the entries of its forwarding database. "
       "The forwarding database is weak to the service.")
   ]
class CIM_TransparentBridgingDynamicForwarding : CIM_Dependency
{
      [Override ("Antecedent"), Min(1), Max(1),
       Description ("The transparent bridging service.")
      ]
   CIM_TransparentBridgingService REF Antecedent;
      [Override ("Dependent"), Weak,
       Description ("Dynamic Forwarding Entry of forwarding database.")
      ]
   CIM_DynamicForwardingEntry REF Dependent;
};

// ==================================================================
//    SwitchPortDynamicForwarding
// ==================================================================
   [Association,
    Description (
       "This association links a dynamic forwarding entry "
       "and the switch port to which the entry applies.")
   ]
class CIM_SwitchPortDynamicForwarding : CIM_Dependency
{
      [Override ("Antecedent"), Min(1), Max(1),
       Description ("The switch port.")
      ]
   CIM_SwitchPort REF Antecedent;
      [Override ("Dependent"),
       Description ("Dynamic Forwarding Entry of forwarding database.")
      ]
```

```
   CIM_DynamicForwardingEntry REF Dependent;
};

// ==================================================================
//     StaticForwardingEntry
// ==================================================================
   [Description (
       "A StaticForwardingEntry represents an entry in the "
       "static (destination-address filtering) database associated "
       "with the switch service.")
   ]
class CIM_StaticForwardingEntry : CIM_LogicalElement
{

      [Propagated("CIM_System.CreationClassName"),
       Key, MaxLen (256),
       Description ("The scoping System's CreationClassName.")
      ]
   string SystemCreationClassName;
      [Propagated("CIM_System.Name"),
       Key, MaxLen (256),
       Description ("The scoping System's Name.")
      ]
   string SystemName;
      [Propagated("CIM_Service.CreationClassName"),
       Key, MaxLen (256),
       Description ("The scoping Service's CreationClassName.")
      ]
   string ServiceCreationClassName;
      [Propagated("CIM_Service.Name"),
       Key, MaxLen (256),
       Description ("The scoping Service's Name.")
      ]
   string ServiceName;
      [Key, MaxLen (256),
       Description (
  "CreationClassName indicates 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.")
      ]
   string CreationClassName;
      [Key, MaxLen (6), Description (
          "Destination MAC address (unicast, multicast or broadcast) "
          "to which the filtering information applies. "
          ""
          "Note that the MAC address is formatted as twelve "
          "hexadecimal digits (e.g., \"010203040506\"), with each "
          "pair representing one of the six octets of the MAC address "
          "in \"canonical\" bit order according to RFC 2469."),
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStaticAddress"}
      ]
   string MACAddress;
      [Description (
          "The status of the entry."),
       ValueMap {"1", "2", "3", "4", "5"},
        Values {"other", "invalid", "permanent", "deleteOnReset",
                "DeleteOnTimeout"},
       Mappingstrings {"MIB.IETF|RFC1493-MIB.dot1dStaticStatus"}
      ]
   uint16 StaticStatus;
      [Description (
          "The ports to which frames with the destination MAC "
          "address arriving on the port represented by the "
          "SwitchPortStaticForwarding association are allowed "
          "to be forwarded.")
      ]
   uint16 AllowedToGo[];
};


// ==================================================================
//    TransparentBridgingStaticForwarding
// ==================================================================
   [Association,
    Description (
       "This association links an instance of the transparent "
       "bridging service to the entries of its static "
       "(destination-address filtering) database.")
   ]
class CIM_SwitchServiceStaticForwarding : CIM_Dependency
{
      [Override ("Antecedent"), Min(1), Max(1),
       Description ("The transparent bridging service.")
      ]
   CIM_TransparentBridgingService REF Antecedent;
      [Override ("Dependent"), Weak,
       Description ("Static Forwarding Entry of forwarding database.")
      ]
   CIM_StaticForwardingEntry REF Dependent;
};

// ==================================================================
//    SwitchPortStaticForwarding
// ==================================================================
   [Association,
    Description (
       "This association links a static database entry "
       "and the switch port to which the entry applies.")
   ]
class CIM_SwitchPortStaticForwarding : CIM_Dependency
{
      [Override ("Antecedent"), Min(1),
       Description ("The switch port.")
      ]
   CIM_SwitchPort REF Antecedent;
      [Override ("Dependent"),
       Description ("Static Forwarding Entry of static database.")
      ]
   CIM_StaticForwardingEntry REF Dependent;
};

[...]

// ==================================================================
```

```
//    SwitchServiceTransparentBridging
// ==================================================================
   [Association, Aggregation, Description (
       "An association linking SwitchService to a component "
       "TransparentBridgingService. The cardinality of the "
       "TransparentBridgingService is 0..1 for a VLAN-unaware "
       "switch.")
   ]
class CIM_SwitchServiceTransparentBridging : CIM_ServiceComponent
{
      [Override ("GroupComponent"), Aggregate, Min(1),
       Description ("The parent service.")
      ]
   CIM_SwitchService REF GroupComponent;
      [Override ("PartComponent"),
       Description ("The component service.")
      ]
   CIM_TransparentBridgingService REF PartComponent;
};

// ==================================================================
//    SwitchServiceSpanningTree
// ==================================================================
   [Association, Aggregation, Description (
       "An association linking SwitchService to a component "
       "SpanningTreeService. The cardinality of the "
       "SpanningTreeService is 0..1 for a VLAN-unaware "
       "switch.")
   ]
class CIM_SwitchServiceSpanningTree : CIM_ServiceComponent
{
      [Override ("GroupComponent"), Aggregate, Min(1),
       Description ("The parent service.")
      ]
   CIM_SwitchService REF GroupComponent;
      [Override ("PartComponent"),
       Description ("The component service.")
      ]
   CIM_SpanningTreeService REF PartComponent;
};

[...]

// ==================================================================
//     VLAN
// ==================================================================
   [Description (
        "An instance of CIM_VLAN represents a VLAN within a switch."
        "In a particular switch, there should be an instance of CIM_VLAN"
        "for every VLAN available in the switch. For example, in a switch"
        "with port-based VLANs, if there are 16 VLANs to which ports can"
        "be assigned (VLAN 1 through VLAN 16), there should be an"
        "instance of CIM_VLAN for each of VLAN 1 through VLAN 16."
        ""
        "CIM_VLAN inherits Name from ServiceAccessPoint. Use this for"
        "the textual name of the VLAN, if there is one. Otherwise,"
        "synthesize a textual name, e.g., VLAN 0003. (Consider leading"
        "zero fill, as shown, to ensure that if the textual VLAN names"
        "are extracted and presented by a management applictions, the"
        "VLAN names will sort in the expected order. It's irritating"
        "to see a list sorted"
        "'VLAN 1, VLAN 10, VLAN 11, VLAN 2, VLAN 3...'"
        "when it could have been"
        "'VLAN 0001, VLAN 0002, ..., VLAN 0009, VLAN 0010, VLAN 0011, ....')"
        "The numeric part of the name should be at least four digits wide"
        "since 802.1Q specifies 4095 VLANs."
        ""
        "It is intended that CIM_VLAN be subclassed only if necessary"
        "to add attributes. The type of the VLAN can be inferred from"
        "the VLANService(s) with which the CIM_VLAN is associated"
        "in the VLANFor association."
        ""
        "An instance of CIM_VLAN may be associated with more than one"
        "VLANService. For example, there are switches that support both"
        "802.1Q VLANs and the vendor's proprietary VLANs. In some such"
        "switches if a broadcast packet received on a port in 802.1Q"
        "VLAN 5 will be transmitted from a port in proprietary VLAN 5."
        "In effect, there is only one VLAN 5, and the type of port"
        "only determines the packet format for tagged packets. In the"
        "case just described, only one instance of CIM_VLAN should be"
        "instantiated for VLAN 5, and it should be associated both with"
        "the 802.1Q VLANService and the proprietary VLANService."
        ""
        "In typical VLAN-aware switches, packets can be assigned to a VLAN"
        "based on the port on which they are received (port-based VLANS),"
        "based on the source MAC address (MAC-based VLANs), or based on"
        "the value of a set of bits in the packet (protocol-based VLANs)."
        "If it is desirable to represent the VLAN assignment predicate"
        "for some MAC-based VLAN switch, it will be necessary to subclass"
        "CIM_VLAN. The list of MAC addresses associated with a VLAN would"
        "be an attribute of the subclass. If it is desirable to represent"
        "the VLAN assignment predicate in a protocol-based VLAN switch,"
        "it will be necessary to subclass CIM_VLAN, CIM_InboundVLAN, or"
        "both. If the predicate applies to all ports in the switch, then"
        "only CIM_VLAN need be subclassed. If the predicate may vary"
        "based on the port, then CIM_InboundVLAN must be subclassed,"
        "and CIM_VLAN might have to be subclassed as well.")
   ]
class CIM_VLAN : CIM_ServiceAccessPoint
{
      [Description (
        "")
      ]
   uint32 VLANNumber;
};


// ==================================================================
//     RelatedTransparentBridgingService
// ==================================================================
   [Association, Description (
        "The association between a VLAN and the forwarding database"
        "(some use the term filtering database) used to determine"
        "which port a packet should be transmitted on, given that it"
        "is assigned to the VLAN and that it has a particular"
        "destination MAC address. The TransparentBridgingService"
```

```
        "represents a forwarding database.") ]

class CIM_RelatedTransparentBridgingService : CIM_ServiceSAPDependency
{
      [Override ("Antecedent"), Min(0)]
   CIM_VLAN  REF Antecedent;
      [Override ("Dependent"), Min(1), Max(1)]
   CIM_TransparentBridgingService REF Dependent;
};


// ==================================================================
//     CIM_RelatedSpanningTree
// ==================================================================
   [Association, Description (
        "This association identifies the spanning tree in which"
        "a forwarding database (TransparentBridgingService) is"
        "nested.") ]
class CIM_RelatedSpanningTree : CIM_ServiceServiceDependency
{
      [Override ("Antecedent"), Min(0)]
   CIM_TransparentBridgingService REF Antecedent;
      [Override ("Dependent"), Min(0), Max(1)]
    CIM_SpanningTreeService REF Dependent;
};


// ==================================================================
//     InboundVLAN
// ==================================================================
   [Association, Description (
        "If there is an association between a particular SwitchPort and"
        "a particular CIM_VLAN, then there is the possibility that"
        "a packet received by the port will be assigned to the VLAN"
        "(or if the packet already has a VLAN tag, that the packet"
        "will not be dropped). If there is no such association,"
        "then there is no possibility that a packet received by the"
        "port will progress through the switch having been assigned"
        "to the VLAN in question.")
   ]

class CIM_InboundVLAN : CIM_SAPSAPDependency
{
      [Override ("Antecedent"), Min(0)]
   CIM_VLAN REF Antecedent;
      [Override ("Dependent"), Min(0)]
   CIM_SwitchPort REF Dependent;

      [Description (
        "If true, packets already tagged with this VLAN number will be"
        "accepted when arriving at this port. For example, if there is"
        "an InboundVLAN association between port 12 and VLAN 7 for"
        "which Tagged is true, then if a packet tagged with VLAN 7"
        "arrives at port 12, the packet will be accepted into the"
        "switch for further processing. If there is no such association,"
        "then the packet will be dropped."
        ""
        "If false, it means that any untagged packets arriving at this"
        "port MIGHT be classified into the associated VLAN. If, for a"
        "particular SwitchPort, there is only one instance of the"
        "association for which Tagged is false, then all incoming"
        "untagged packets will be classified into that VLAN. This is"
        "the typical configuration of a non-trunk port in a switch"
        "implementing port-based VLANs. If there is more than one such"
        "association instance, then the packet MIGHT be classified"
        "into any one of them, based on some criterion other than the"
        "identity of the switch port. For example, in a MAC-based VLAN"
        "switch, the decision would be based on the source MAC address."
        "In a protocol-based VLAN switch, the decision would be based"
        "on the values of some set of bits in the packet. "
        ""
        "Note that the MAC address is formatted as twelve "
        "hexadecimal digits (e.g., \"010203040506\"), with each "
        "pair representing one of the six octets of the MAC address "
        "in \"canonical\" bit order according to RFC 2469.") ]
   boolean Tagged;

      [Description (
        "Default should be true if untagged packets received by the"
        "Antecedent SwitchPort are assigned to the Dependent VLAN"
        "by default. For 802.1Q-compliant ports, Default should be"
        "true on the association instance connecting a SwitchPort to"
        "the CIM_VLAN corresponding to the port's PVID. Default"
        "should never be true if Tagged is true -- it applies only"
        "to untagged packets.") ]
   boolean Default;
};


// ==================================================================
//     OutboundVLAN
// ==================================================================
   [Association, Description (
        "If there is no instance of OutboundVLAN between a given"
        "SwitchPort and CIM_VLAN, then any packet that has been"
        "assigned to the VLAN and whose destination address is"
        "associated with the port will be dropped by the switch"
        "without being transmitted. Otherwise, the packet will be"
        "transmitted.")
   ]
class CIM_OutboundVLAN : CIM_SAPSAPDependency
{
      [Override ("Antecedent"), Min(0)]
   CIM_VLAN REF Antecedent;
      [Override ("Dependent"), Min(0)]
   CIM_SwitchPort REF Dependent;

      [Description (
        "If Tagged is true, then the packet will be transmitted in"
        "encapsulated form, tagged with the associated VLAN tag. If"
        "Tagged is false, the packet will be trasmitted without any"
        "VLAN tag.") ]
   boolean Tagged;
};

// ==================================================================
```

```
//     VLANService
// ==================================================================
   [Abstract,
    Description (
        "VLANService represents the VLAN aspects of the function"
        "performed by a switch. Some VLAN-aware devices participate"
        "in protocols where VLAN information is propagated among"
        "switches, e.g., GVRP in 802.1Q switches and VTP in Cisco"
        "Catalyst switches. VLANService also represents the function"
        "performed by the switch as a participant in such a protocol."
        ""
        "VLANService must be subclassed so that instances can be"
        "distinguished by their class. If there is a name assigned to"
        "a set of VLAN-aware switches, e.g., the VTP domain name in"
        "Cisco Catalyst switches, use the Name attribute inherited"
        "from CIM_Service to store the name."
        ""
        "A VLANService should be instantiated in a VLAN-aware switch"
        "even if there is no GVRP-like protocol.") ]
class CIM_VLANService : CIM_Service
{
};


// ==================================================================
//     802dot1QVLANService
// ==================================================================
   [Description (
        "If a switch supports 802.1Q, an instance of this class should"
        "be instantiated in the switch. If the switch supports GVRP,"
        "this class represents the function that the switch performs"
        "with respect to GVRP.") ]
class CIM_802dot1QVLANService : CIM_VLANService
{
};


// ==================================================================
//     SwitchServiceVLAN
// ==================================================================
   [Association, Aggregation, Description (
       "An association linking SwitchService to a component "
       "VLANService.")
   ]
class CIM_SwitchServiceVLAN : CIM_ServiceComponent
{
      [Override ("GroupComponent"), Min(1), Max(1), Aggregate]
   CIM_SwitchService REF GroupComponent;
      [Override ("PartComponent")]
   CIM_VLANService REF PartComponent;
};


// ==================================================================
//     VLANFor
// ==================================================================
   [Association]
class CIM_VLANFor : CIM_ServiceSAPDependency
{
      [Override ("Antecedent"), Min(0)]
   CIM_VLAN REF Antecedent;
      [Override ("Dependent"), Min(1)]
   CIM_VLANService REF Dependent;
};

[...]

// ==================================================================
// end of file
// ==================================================================
```

B.5 CIM Physical Schema

```
// ==================================================================
// Title:       Physical MOF Specification 2.2
// Filename:    CIM_Physical22.mof
// Author:      DMTF System and Devices Working Group
// Last Change: 06/09/1999
// ==================================================================

[...]

// ==================================================================
// PhysicalLink
// ==================================================================
   [Description (
 "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.")
   ]
class CIM_PhysicalLink : CIM_PhysicalElement
{
[Description ("The maximum length of the PhysicalLink in feet."),
 Units ("Feet")
]
   real64 MaxLength;
[Description (
  "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."),
 Units ("Feet")
]
   real64 Length;
[Description (
  "Boolean indicating whether the PhysicalLink is an actual "
  "cable (TRUE) or a wireless connection (FALSE).")
]
   boolean Wired;
[Description("The MediaType property defines the particular "
  "type of Media through which transmission signals pass. "
  "Common network media include twisted-pair (value=11 or 12), "
  "coaxial (7, 8 or 9) and fiber-optic cable (10)."),
      Values {"Unknown", "Other", "Cat1", "Cat2", "Cat3", "Cat4",
            "Cat5", "50-ohm Coaxial", "75-ohm Coaxial",
            "100-ohm Coaxial", "Fiber-optic", "UTP", "STP",
            "Ribbon Cable", "Twinaxial", "Optical 9um",
            "Optical 50um", "Optical 62.5um"}
]
   uint16 MediaType;
};

// ==================================================================
// ElementsLinked
// ==================================================================
   [Association,
    Description (
 "The ElementsLinked association indicates which Physical"
 "Elements are cabled together by a PhysicalLink.")
   ]
class CIM_ElementsLinked : CIM_Dependency
{
[Override ("Antecedent"),
 Description ("The PhysicalLink.")
]
   CIM_PhysicalLink REF Antecedent;
[Override ("Dependent"),
 Description ("The PhysicalElement that is linked.")
]
   CIM_PhysicalElement REF Dependent;
};

// ==================================================================
// PhysicalConnector
// ==================================================================
   [Description (
 "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.")
   ]
class CIM_PhysicalConnector : CIM_PhysicalElement
{
[Description (
  "A free-form string describing the pin configuration and "
  "signal usage of a PhysicalConnector.")
]
   string ConnectorPinout;
[Description (
  "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 {"Unknown", "Other", "Male", "Female", "Shielded",
            "Unshielded", "SCSI (A) High-Density (50 pins)",
            "SCSI (A) Low-Density (50 pins)",
            "SCSI (P) High-Density (68 pins)", "SCSI SCA-I (80 pins)",
            "SCSI SCA-II (80 pins)", "Fibre Channel (DB-9, Copper)",
            "Fibre Channel (Optical Fibre)",
            "Fibre Channel SCA-II (40 pins)",
            "Fibre Channel SCA-II (20 pins)", "Fibre Channel BNC",
            "ATA 3-1/2 Inch (40 pins)", "ATA 2-1/2 Inch (44 pins)",
            "ATA-2", "ATA-3", "ATA/66", "DB-9", "DB-15", "DB-25",
            "DB-36", "RS-232C", "RS-422", "RS-423", "RS-485", "RS-449",
            "V.35", "X.21", "IEEE-488", "AUI", "UPT Category 3",
            "UPT Category 4", "UPT Category 5", "BNC", "RJ11", "RJ45",
            "Fiber MIC", "Apple AUI", "Apple GeoPort", "PCI", "ISA",
```

```
             "EISA", "VESA", "PCMCIA", "PCMCIA Type I",
             "PCMCIA Type II", "PCMCIA Type III", "ZV Port", "CardBus",
             "USB", "IEEE 1394", "HIPPI", "HSSDC (6 pins)", "GBIC",
             "DIN", "Mini-DIN", "Micro-DIN", "PS/2", "Infrared",
             "HP-HIL", "Access.bus", "NuBus", "Centronics",
             "Mini-Centronics", "Mini-Centronics Type-14",
             "Mini-Centronics Type-20", "Mini-Centronics Type-26",
             "Bus Mouse", "ADB", "AGP", "VME Bus", "VME64",
             "Proprietary", "Proprietary Processor Card Slot",
             "Proprietary Memory Card Slot", "Proprietary I/O Riser Slot",
             "PCI-66MHZ", "AGP2X", "AGP4X", "PC-98", "PC-98-Hireso",
             "PC-H98", "PC-98Note", "PC-98Full", "SSA SCSI",
             "Circular", "On Board IDE Connector",
             "On Board Floppy Connector", "9 Pin Dual Inline",
             "25 Pin Dual Inline", "50 Pin Dual Inline",
             "68 Pin Dual Inline", "On Board Sound Connector",
             "Mini-jack", "PCI-X", "Sbus IEEE 1396-1993 32 bit",
             "Sbus IEEE 1396-1993 64 bit", "MCA", "GIO", "XIO",
             "HIO", "NGIO", "PMC", "MTRJ", "VF-45", "Future I/O",
             "SC", "SG", "Electrical", "Optical", "Ribbon", "GLM",
             "1x9", "Mini SG", "LC", "HSSC"}
]
   uint16 ConnectorType[];
};

// ==================================================================
// ConnectedTo
// ==================================================================
   [Association,
    Description ("The ConnectedTo association indicates that two or "
       "more PhysicalConnectors are connected together.")
   ]
class CIM_ConnectedTo : CIM_Dependency
{
      [Override ("Antecedent"),
       Description ("The Antecedent represents a PhysicalConnector "
        "that serves as one end of the connection.")
      ]
   CIM_PhysicalConnector REF Antecedent;
      [Override ("Dependent"),
       Description (
  "The Dependent represents another PhysicalConnector "
        "that serves as the other end of the connection.")
      ]
   CIM_PhysicalConnector REF Dependent;
};

[...]

// =================================================================
// LinkHasConnector
// =================================================================
   [Association,
    Aggregation,
    Description (
 "Cables and Links utilize PhysicalConnectors to actually "
 "'connect' PhysicalElements. This association explicitly "
 "defines this relationship of Connectors for PhysicalLinks.")
   ]
class CIM_LinkHasConnector : CIM_Component
{
[Override ("GroupComponent"),
 Aggregate, Max (1),
 Description ("The PhysicalLink that has a Connector.")
]
   CIM_PhysicalLink REF GroupComponent;
[Override ("PartComponent"),
 Description ("The PhysicalConnector.")
]
   CIM_PhysicalConnector REF PartComponent;
};

[...]

// ==================================================================
// AdapterActiveConnection
// ==================================================================
   [Association,
    Description (
       "The AdapterActiveConnection relationship indicates that a "
       "NetworkAdapter is using the referenced PhysicalConnector "
       "to output to the network. This relationship is important when "
       "the Adapter can choose to output from one of several "
       "Connectors. The Connectors may be associated with the "
       "NetworkAdapter in a Realizes relationship - but this is not "
       "required. This association provides additional information "
       "(i.e., 'in use for communication') different than Realizes.")
   ]
class CIM_AdapterActiveConnection : CIM_Dependency
{
      [Override ("Antecedent"),
       Description ("The PhysicalConnector.")
      ]
   CIM_PhysicalConnector REF Antecedent;
      [Override ("Dependent"),
       Description (
        "The NetworkAdapter that transmits using the Connector.")
      ]
   CIM_NetworkAdapter REF Dependent;
};

[...]

// =================================================================
// end of file
// =================================================================
```

B.6 CIM System Schema

```
// ==================================================================
// Title:    System MOF specification 2.2
// Filename:     CIM_System22.mof
// Author:       DMTF System and Devices Working Group
// Last Change:  06/08/1999 - Version 2.2
// ==================================================================

[...]

// ==================================================================
// UnitaryComputerSystem
// ==================================================================
   [Description (
       "A class derived from ComputerSystem that represents a Desktop, "
       "Mobile, NetPC, Server or other type of a single node Computer "
       "System.")
   ]
class CIM_UnitaryComputerSystem : CIM_ComputerSystem
{
      [Description (
  "This object contains the data needed to find either the "
  "initial load device (its key) or the boot service to "
  "request the operating system to start up. In addition, "
  "the load parameters (ie, a pathname and parameters) "
  "may also be specified.")
      ]
   string InitialLoadInfo[];
      [Description (
  "This object contains the data identifying either the "
  "initial load device (its key) or the boot service that "
  "requested the last operating system load. In addition, "
  "the load parameters (ie, a pathname and parameters) "
  "may also be specified."),
 MappingStrings {
  "MIB.IETF|HOST-RESOURCES-MIB.hrSystemInitialLoadDevice",
  "MIB.IETF|HOST-RESOURCES-MIB.hrSystemInitialLoadParameters"}
      ]
   string LastLoadInfo;
      [Description (
  "If enabled (value = 4), the UnitaryComputerSystem can be "
  "reset via hardware (e.g. the power and reset buttons). If "
    "disabled (value = 3), hardware reset is not allowed. In "
  "addition to Enabled and Disabled, other Values for the "
  "property are also defined - \"Not Implemented\" (5), "
  "\"Other\" (1) and \"Unknown\" (2)."),
       ValueMap {"1", "2", "3", "4", "5"},
 Values {"Other", "Unknown", "Disabled",
       "Enabled", "Not Implemented"},
       MappingStrings {"MIF.DMTF|System Hardware Security|001.4"}
      ]
   uint16 ResetCapability;
      [Description (
  "Boolean indicating that the ComputerSystem, with its "
  "running OperatingSystem, supports power management. This "
  "boolean does not indicate 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.")
    ]
 boolean PowerManagementSupported;
    [Description(
"Indicates the specific power-related capabilities of a "
"ComputerSystem and its associated running OperatingSystem. "
"The values, 0=\"Unknown\", 1=\"Not Supported\", and "
"2=\"Disabled\" are self-explanatory. "
"The value, 3=\"Enabled\" indicates 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 system can change its power state based on usage or "
"other criteria. \"Power State Settable\" (5) indicates that "
"the SetPowerState method is supported. \"Power Cycling "
     "Supported\" (6) indicates that the SetPowerState method "
"can be invoked with the PowerState input variable set to 5 ("
"\"Power Cycle\"). \"Timed Power On Supported\" (7) indicates "
"that the SetPowerState method can be invoked with the Power"
"State input variable set to 5 (\"Power Cycle\") and the Time "
"parameter set to a specific date and time, or interval, "
"for power-on."),
     Values {"Unknown", "Not Supported", "Disabled",
     "Enabled", "Power Saving Modes Entered Automatically",
     "Power State Settable", "Power Cycling Supported",
     "Timed Power On Supported"},
     MappingStrings {"MIF.DMTF|System Power Controls|001.2"}
    ]
 uint16 PowerManagementCapabilities[];
    [Description(
      "Indicates the current power state of the ComputerSystem and "
      "its associated OperatingSystem. Regarding the Power Save "
      "states, these are defined as follows: Value 4 (\"Power Save "
      "- Unknown\") indicates that the System is known to be in a "
      "power save mode, but its exact status in this mode is "
      "unknown; 2 (\"Power Save - Low Power Mode\") indicates that "
      "the System is in a power save state but still functioning, "
      "and may exhibit degraded performance; 3 (\"Power Save - "
      "Standby\") describes that the System is not functioning but "
      "could be brought to full power 'quickly'; value 7 (\"Power Save "
      "- Warning\") indicates that the ComputerSystem is in a warning "
      "state, though also in a power save mode; and, values 8 and 9 "
      "describe the ACPI \"Hibernate\" and \"Soft Off\" states."),
     Values {"Unknown", "Full Power", "Power Save - Low Power Mode",
           "Power Save - Standby", "Power Save - Unknown",
           "Power Cycle", "Power Off", "Power Save - Warning",
           "Power Save - Hibernate", "Power Save - Soft Off"}
    ]
 uint16 PowerState;
    [Description (
      "The event that caused the System to power up. This "
      "information is available in SMBIOS, in the Type 1 structure, "
      "the Wake Up Type attribute."),
     Values {"Reserved", "Other", "Unknown", "APM Timer",
           "Modem Ring", "LAN Remote", "Power Switch", "PCI PME#",
```

```
            "A/C Power Restored"}
      ]
   uint16 WakeUpType;
      [Description(
        "SetPowerState defines the desired power state of a "
        "ComputerSystem and its running OperatingSystem, and when "
        "the system should be put into that state. The PowerState "
        "parameter is specified as one of the following integer "
        "values: 1=\"Full Power\", 2=\"Power Save - Low Power Mode\", "
        "3=\"Power Save - Standby\", 4=\"Power Save - Other\", "
        "5=\"Power Cycle\", 6=\"Power Off\", 7=\"Hibernate\" and "
        "8=\"Soft Off\". The Time parameter (for all state changes "
        "but 5, \"Power Cycle\") indicates when the power state "
        "should be set, either as a regular date-time value or as "
        "an interval value (where the interval begins when the "
        "method invocation is received). When the PowerState "
        "parameter is equal to 5, \"Power Cycle\", the Time "
        "parameter indicates when the system should power on "
        "again. Power off is immediate. SetPowerState should "
        "return 0 if successful, 1 if the specified State and "
        "Time request is not supported, and some other value if "
        "any other error occurred. In a subclass, the "
        "set of possible return codes could be specified, using a "
        "ValueMap qualifier on the method. The strings to which the "
        "ValueMap contents are 'translated' may also be specified in "
        "the subclass as a Values array qualifier.")
      ]
   uint32 SetPowerState(
    [IN, ValueMap {"1", "2", "3", "4", "5", "6", "7", "8"},
         Values {"Full Power", "Power Save - Low Power Mode",
          "Power Save - Standby", "Power Save - Other",
          "Power Cycle", "Power Off", "Hibernate", "Soft Off"}]
         uint16 PowerState,
    [IN] datetime Time);
};

[...]

// ==================================================================
// VirtualComputerSystem
// ==================================================================
   [Description (
       "A class derived from ComputerSystem that represents the "
       "ability to virtualize or emulate another ComputerSystem.")
   ]
class CIM_VirtualComputerSystem : CIM_ComputerSystem
{
      [Description (
        "A string describing the type of System or hardware platform "
  "that is virtualized. OperatingSystem information is obtained "
  "via the RunningOS inherited from ComputerSystem.")
      ]
   string VirtualSystem;
};

[...]

// ==================================================================
// Cluster
// ==================================================================
   [Description (
       "A class derived from ComputerSystem that 'is made up of' two "
       "or more ComputerSystems which operate together as an atomic, "
       "functional whole to increase the performance, resources and/or "
       "RAS (Reliability, Availability and Serviceability) of the "
       "component ComputerSystems, related to some aspects of these "
       "ComputerSystems.")
   ]
class CIM_Cluster : CIM_ComputerSystem
{
      [Description (
  "Interconnect is a free form string that describes the "
  "interconnection mechanism for the Cluster.")
      ]
   string Interconnect;
      [Description (
  "InterconnectAddress indicates the address of the Cluster "
  "System, which is dependent on the interconnection scheme. "
  "If no address is available or applicable, a null string "
  "should be used.")
      ]
   string InterconnectAddress;
      [Description (
  "The cluster types. This specifies whether the cluster is for "
  "failover (value=2), performance (3), etc. The values which "
  "can be specified are not mutually exclusive. Thus, Types is "
  "an array."),
       Values {"Unknown", "Other", "Failover",
       "Performance", "Distributed OS", "Node Grouping",
             "SysPlex"}
      ]
   uint16 Types[];
      [Description (
  "Indicates the maximum number of nodes that may participate in "
  "the Cluster.  If unlimited, enter 0.")
      ]
   uint32 MaxNumberOfNodes;
      [Description (
  "Indicates the state of the Cluster.  The cluster can be "
  "defined to be on-line (value=2), off-line (3), in a degraded "
  "mode of operation (4) or unavailable (5)."),
       Values {"Unknown", "Other", "On-line", "Off-line",
       "Degraded", "Unavailable"}
      ]
   uint16 ClusterState;
};

[...]

// ==================================================================
// ClusteringService
// ==================================================================
   [Description (
       "ClusteringService represents the functionality provided by a "
       "Cluster. For example, failover functionality may be modeled as "
       "a Service of a failover Cluster.")
```

```
   ]
class CIM_ClusteringService : CIM_Service
{
      [Description (
  "AddNode brings a new ComputerSystem into a Cluster. "
  "The node to be added is specified as a parameter to the "
  "method. The return value should be 0 if the Computer"
  "System is successfully added, 1 if the method is not "
  "supported and any other number if an error occurred. "
        "In a subclass, the set of possible return codes could be "
        "specified, using a ValueMap qualifier on the method. The "
        "strings to which the ValueMap contents are 'translated' "
        "may also be specified in the subclass as a Values array "
        "qualifier.")
      ]
   uint32 AddNode ([IN] CIM_ComputerSystem ref CS);
      [Description (
  "EvictNode removes a ComputerSystem from a Cluster. "
  "The node to be evicted is specified as a parameter to the "
  "method. The return value should be 0 if the ComputerSystem "
  "is successfully evicted, 1 if the method is not supported "
  "and any other number if an error occurred. In a subclass, "
        "the set of possible return codes could be specified, using a "
        "ValueMap qualifier on the method. The strings to which the "
        "ValueMap contents are 'translated' may also be specified in "
        "the subclass as a Values array qualifier.")
      ]
   uint32 EvictNode ([IN] CIM_ComputerSystem ref CS);
};

// ==================================================================
// ClusteringSAP
// ==================================================================
   [Description (
       "A representation of the access points of a ClusteringService.")
   ]
class CIM_ClusteringSAP : CIM_ServiceAccessPoint
{
};

// ==================================================================
// ClusterServiceAccessBySAP
// ==================================================================
   [Association,
    Description (
       "ClusterServiceAccessBySAP is the relationship between a "
       "ClusteringService and its access points.")
   ]
class CIM_ClusterServiceAccessBySAP : CIM_ServiceAccessBySAP
{
      [Override ("Antecedent"),
       Description ("The ClusteringService.")
      ]
   CIM_ClusteringService REF Antecedent;
      [Override ("Dependent"),
       Description ("An access point for the ClusteringService.")
      ]
   CIM_ClusteringSAP REF Dependent;
};

// ==================================================================
// HostedClusterService
// ==================================================================
   [Association,
    Description (
       "HostedClusterService defines the hosting Cluster for a "
       "ClusteringService.  Since this relationship is subclassed "
       "from HostedService, it inherits the scoping/naming scheme "
       "defined for Service - where a Service is weak to its "
       "hosting System.  In this case, a ClusteringService must be "
       "weak to its hosting Cluster System.")
   ]
class CIM_HostedClusterService : CIM_HostedService
{
      [Override ("Antecedent"),
 Description ("The Cluster.")
      ]
   CIM_Cluster REF Antecedent;
      [Override ("Dependent"),
 Description (
  "The ClusteringService that is hosted on the Cluster.")
      ]
   CIM_ClusteringService REF Dependent;
};

// ==================================================================
// HostedClusterSAP
// ==================================================================
   [Association,
    Description (
       "HostedClusterSAP defines the hosting ComputerSystem for a "
       "ClusteringSAP. Since this relationship is subclassed "
       "from HostedAccessPoint, it inherits the scoping/naming scheme "
       "defined for AccessPoint - where an AccessPoint is weak to "
       "its hosting System.  In this case, a ClusteringSAP must be "
       "weak to its hosting ComputerSystem.")
   ]
class CIM_HostedClusterSAP : CIM_HostedAccessPoint
{
      [Override ("Antecedent"),
 Description ("The ComputerSystem.")
      ]
   CIM_ComputerSystem REF Antecedent;
      [Override ("Dependent"),
 Description (
  "The ClusteringSAP that is hosted on the ComputerSystem.")
      ]
   CIM_ClusteringSAP REF Dependent;
};

[...]

// ==================================================================
// end of file
// ==================================================================
```

Anhang C

Generierter IDL-Code

```
// StP
// Created on Fri Aug 13 13:12:39 1999 for fischmax@hpheger2
//   from system /proj/atm/fischmax/StP/atm_fischmax

#ifndef _CAE_idl_
#define _CAE_idl_

typedef unsigned short      uint16
typedef unsigned long       uint32
typedef unsigned long long  uint64
typedef string              datetime


// stp class declarations
interface CIM_ManagedSystemElement;
interface CIM_LogicalElement;
interface CIM_Service;
interface CIM_ServiceAccessPoint;
interface CAE_compObjectService;
interface CIM_ProtocolEndpoint;
interface CIM_CollectionOfMSEs;
interface CIM_NetworkService;
interface CIM_LogicalNetwork;
interface CAE_ServerService;
interface CIM_LogicalDevice;
interface CAE_VirtualNetwork;
interface CAE_LE_ServerService;
interface CIM_NetworkAdapter;
interface CAE_VirtualNetworkService;
interface CIM_IPProtocolEndpoint;
interface CIM_System;
interface CAE_ClientService;
interface CIM_ForwardingService;
interface CIM_LANEndpoint;
interface CIM_VLANService;
interface CIM_IPSubnet;
interface CAE_ELAN;
interface CAE_ATMEndpoint;
interface CIM_PhysicalElement;
interface CAE_VLAN;
interface CAE_LECEndpoint;
interface CAE_LES_Service;
interface CIM_VLAN;
interface CIM_EthernetAdapter;
interface CIM_SwitchPort;
interface CAE_ATMARPServerService;
interface CIM_AdminDomain;
interface CIM_SwitchService;
interface CAE_BUS_Service;
interface CAE_LECS_Service;
interface CIM_IPOAClientEndpoint;
interface CIM_TokenRingAdapter;
interface CIM_802dot1QVLANService;
interface CAE_VirtualGroupingService;
interface CAE_LEC_Service;
interface CAE_VLANService;
interface CAE_ELANService;
interface CIM_ComputerSystem;
interface CAE_LISService;
interface CAE_IPOA_ClientService;
interface CAE_ATMAdapter;
interface CAE_LIS;
interface CIM_LANSegment;
interface CAE_IPOAClientEndpoint;
interface CIM_ServiceServiceDependency;
interface CIM_BindsToLANEndpoint;
interface CIM_ActiveConnection;
interface CIM_LogicalNetworkService;
interface CIM_PhysicalConnector;
// end stp class declarations

// stp class definition 2
interface CIM_ManagedSystemElement
{
// stp class members
    attribute string Description;
    attribute string Name;
    attribute string Caption;
    attribute datetime InstallDate;
    attribute string Status;
// end stp class members
};
// end stp class definition

// stp class definition 4
interface CIM_LogicalElement : CIM_ManagedSystemElement
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 8
interface CIM_Service : CIM_LogicalElement
{
// stp class members
    attribute string Name;
    attribute string StartMode;
    attribute string CreationClassName;
    readonly attribute boolean Started;
    attribute string SystemName;
    attribute string SystemCreationClassName;
    uint32 StartService(void);
    uint32 StopService(void);
// end stp class members
};
// end stp class definition

// stp class definition 11
interface CIM_ServiceAccessPoint : CIM_LogicalElement
{
// stp class members
    attribute string Name;
    attribute string CreationClassName;
    attribute string SystemCreationClassName;
    attribute string SystemName;
// end stp class members
};
// end stp class definition

// stp class definition 54
interface CAE_compObjectService : CIM_Service
{
// stp class members
    attribute string UsageState;
    attribute string AdminState;
```

```
    attribute datetime Uptime;
    uint32 lock(void);
    uint32 unlock(void);
// end stp class members
};
// end stp class definition

// stp class definition 21
interface CIM_ProtocolEndpoint : CIM_ServiceAccessPoint
{
// stp class members
    attribute string NameFormat;
    attribute string Type;
    attribute string Name;
    attribute string OtherTypeDescription;
    attribute uint64 MaxSpeed;
    attribute uint64 Speed;
// end stp class members
};
// end stp class definition

// stp class definition 17
interface CIM_CollectionOfMSEs
{
// stp class members
    attribute string CollectionID;
    attribute string Description;
    attribute string Caption;
// end stp class members
};
// end stp class definition

// stp class definition 19
interface CIM_NetworkService : CIM_Service
{
// stp class members
    attribute string ServiceURL;
    attribute sequence<string> Keywords;
    attribute sequence<string> StartupConditions;
    attribute sequence<string> StartupParameters;
// end stp class members
};
// end stp class definition

// stp class definition 20
interface CIM_LogicalNetwork : CIM_CollectionOfMSEs
{
// stp class members
    attribute string SystemName;
    attribute string CreationClassName;
    attribute string SystemCreationClassName;
    attribute string Name;
    attribute string OtherTypeDescription;
    attribute string Type;
// end stp class members
};
// end stp class definition

// stp class definition 57
interface CAE_ServerService : CAE_compObjectService
{
// stp class members
    attribute uint32 Accept;
    attribute uint32 Bad;
    attribute uint32 Request;
    attribute uint32 Error;
    attribute uint32 Utilization;
    attribute datetime Delay;
    uint32 listClients(void);
// end stp class members
};
// end stp class definition

// stp class definition 5
interface CIM_LogicalDevice : CIM_LogicalElement
{
// stp class members
    attribute string DeviceID;
    attribute boolean PowerManagementSupported;
    attribute sequence<uint16> PowerManagementCapabilities;
    attribute uint16 Availability;
    attribute uint16 StatusInfo;
    attribute uint32 LastErrorCode;
    attribute string CreationClassName;
    attribute string ErrorDescription;
    attribute string SystemName;
    attribute boolean ErrorCleared;
    attribute sequence<string> OtherIdentifyingInfo;
    attribute sequence<string> IdentifyingDescriptions;
    attribute uint64 PowerOnHours;
    attribute uint64 TotalPowerOnHours;
    attribute string SystemCreationClassName;
    uint32 SetPowerState(uint16 PowerState, datetime Time);
    uint32 Reset(void);
    uint32 EnableDevice(boolean Enabled);
// end stp class members
};
// end stp class definition

// stp class definition 46
interface CAE_VirtualNetwork : CIM_LogicalNetwork
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 65
interface CAE_LE_ServerService : CAE_ServerService
{
// stp class members
    attribute string PrimaryATMAddress;
    uint32 showErrorLog(void);
    uint32 clearErrorLog(void);
// end stp class members
};
// end stp class definition

// stp class definition 36
interface CIM_NetworkAdapter : CIM_LogicalDevice
{
// stp class members
    attribute sequence<string> NetworkAddresses;
    attribute uint64 Speed;
    attribute uint64 MaxSpeed;
    attribute string PermanentAddress;
    attribute boolean FullDuplex;
    attribute uint64 OctetsReceived;
    attribute boolean AutoSense;
    attribute uint64 OctetsTransmitted;
// end stp class members
};
// end stp class definition

// stp class definition 49
interface CAE_VirtualNetworkService : CIM_NetworkService
{
// stp class members
    uint32 excludeMember(void);
    uint32 includeMember(void);
    uint32 checkAvailibility(void);
    uint32 listMembers(void);
// end stp class members
};
// end stp class definition

// stp class definition 24
interface CIM_IPProtocolEndpoint : CIM_ProtocolEndpoint
{
// stp class members
    attribute string SubnetMask;
    attribute string Address;
    attribute uint16 AddressType;
    attribute uint16 IPVersionSupport;
// end stp class members
};
// end stp class definition

// stp class definition 6
interface CIM_System : CIM_LogicalElement
{
// stp class members
    attribute string Name;
    attribute string NameFormat;
    attribute string CreationClassName;
    attribute string PrimaryOwnerName;
    attribute sequence<string> Roles;
    attribute string PrimaryOwnerContact;
// end stp class members
};
// end stp class definition

// stp class definition 58
interface CAE_ClientService : CAE_compObjectService
{
// stp class members
    attribute uint32 Request;
    attribute uint32 Accept;
    attribute datetime Delay;
// end stp class members
};
// end stp class definition

// stp class definition 32
```

```
interface CIM_ForwardingService : CIM_NetworkService
{
// stp class members
    attribute uint16 ProtocolType;
    attribute string OtherProtocolType;
// end stp class members
};
// end stp class definition

// stp class definition 25
interface CIM_LANEndpoint : CIM_ProtocolEndpoint
{
// stp class members
    attribute uint16 LANType;
    attribute string OtherLANType;
    attribute string LANID;
    attribute string MACAddress;
    attribute uint32 MaxDataSize;
    attribute sequence<string> AliasAddresses;
    attribute sequence<string> GroupAddresses;
// end stp class members
};
// end stp class definition

// stp class definition 0
interface CIM_VLANService : CIM_Service
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 22
interface CIM_IPSubnet : CIM_LogicalNetwork
{
// stp class members
    attribute string SubnetNumber;
    attribute string SubnetMask;
    attribute uint16 AddressType;
// end stp class members
};
// end stp class definition

// stp class definition 48
interface CAE_ELAN : CAE_VirtualNetwork
{
// stp class members
    attribute uint16 ELANType;
    attribute string ELANName;
// end stp class members
};
// end stp class definition

// stp class definition 43
interface CAE_ATMEndpoint : CIM_ProtocolEndpoint
{
// stp class members
    attribute string ATMAddress;
    attribute uint16 AddressType;
// end stp class members
};
// end stp class definition

// stp class definition 3
interface CIM_PhysicalElement : CIM_ManagedSystemElement
{
// stp class members
    attribute string Tag;
    attribute string Manufacturer;
    attribute string Model;
    attribute string SKU;
    attribute string CreationClassName;
    attribute string SerialNumber;
    attribute boolean PoweredOn;
    attribute string Version;
    attribute string PartNumber;
    attribute string OtherIdentifyingInfo;
// end stp class members
};
// end stp class definition

// stp class definition 47
interface CAE_VLAN : CAE_VirtualNetwork
{
// stp class members
    attribute uint32 VLANID;
// end stp class members
};
// end stp class definition

// stp class definition 44
interface CAE_LECEndpoint : CIM_LANEndpoint
{
// stp class members
    attribute uint16 ELANType;
// end stp class members
};
// end stp class definition

// stp class definition 73
interface CAE_LES_Service : CAE_LE_ServerService
{
// stp class members
    attribute string ELANName;
    attribute uint16 ELANType;
    attribute uint32 MaxFrameSize;
    uint32 flushCache(void);
// end stp class members
};
// end stp class definition

// stp class definition 35
interface CIM_VLAN : CIM_ServiceAccessPoint
{
// stp class members
    attribute uint32 VLANNumber;
// end stp class members
};
// end stp class definition

// stp class definition 38
interface CIM_EthernetAdapter : CIM_NetworkAdapter
{
// stp class members
    attribute uint32 MaxDataSize;
    attribute sequence<uint16> Capabilities;
    attribute sequence<string> CapabilityDescriptions;
    attribute sequence<string> EnabledCapabilities;
    attribute uint32 SymbolErrors;
    attribute uint64 TotalPacketsTransmitted;
    attribute uint64 TotalPacketsReceived;
    attribute uint32 AlignmentErrors;
    attribute uint32 FCSErrors;
    attribute sequence<string> NetworkAddresses;
    attribute uint32 SingleCollisionFrames;
    attribute uint32 FrameTooLongs;
    attribute uint32 MultipleCollisionFrames;
    attribute uint32 SQETestErrors;
    attribute uint32 DeferredTransmissions;
    attribute uint32 LateCollisions;
    attribute uint32 ExcessiveCollisions;
    attribute uint32 InternalMACTransmitErrors;
    attribute uint32 InternalMACReceiveErrors;
    attribute uint32 CarrierSenseErrors;
// end stp class members
};
// end stp class definition

// stp class definition 34
interface CIM_SwitchPort : CIM_ProtocolEndpoint
{
// stp class members
    attribute uint16 PortNumber;
// end stp class members
};
// end stp class definition

// stp class definition 68
interface CAE_ATMARPServerService : CAE_ServerService
{
// stp class members
    attribute string PrimaryATMAddress;
    uint32 flushARPCache(void);
    uint32 listClients(void);
// end stp class members
};
// end stp class definition

// stp class definition 18
interface CIM_AdminDomain : CIM_System
{
// stp class members
    attribute string NameFormat;
// end stp class members
};
// end stp class definition

// stp class definition 33
interface CIM_SwitchService : CIM_ForwardingService
{
// stp class members
    attribute string BridgeAddress;
    attribute uint16 NumPorts;
    attribute uint16 BridgeType;
```

```
// end stp class members
};
// end stp class definition

// stp class definition 74
interface CAE_BUS_Service : CAE_LE_ServerService
{
// stp class members
    attribute string ELANName;
// end stp class members
};
// end stp class definition

// stp class definition 72
interface CAE_LECS_Service : CAE_LE_ServerService
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 0
interface CIM_IPOAClientEndpoint : CIM_IPProtocolEndpoint
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 39
interface CIM_TokenRingAdapter : CIM_NetworkAdapter
{
// stp class members
    attribute uint32 MaxDataSize;
    attribute sequence<uint16> Capabilities;
    attribute sequence<string> CapabilityDescriptions;
    attribute sequence<uint16> EnabledCapabilities;
    attribute uint32 RingStatus;
    attribute uint16 RingState;
    attribute uint16 RingOpenStatus;
    attribute uint16 RingSpeed;
    attribute uint32 BurstErrors;
    attribute uint32 ACErrors;
    attribute uint32 AbortTransErrors;
    attribute uint32 InternalErrors;
    attribute sequence<string> NetworkAddresses;
    attribute uint32 LostFrameErrors;
    attribute uint32 FrequencyErrors;
    attribute uint32 ReceiveCongestions;
    attribute uint32 FrameCopiedErrors;
    attribute uint32 TokenErrors;
    attribute uint32 SoftErrors;
    attribute uint32 HardErrors;
    attribute uint32 SignalLossCount;
    attribute uint32 TransmittedBeacons;
    attribute uint32 Recoverys;
    attribute uint32 LobeWires;
    attribute uint32 Removes;
    attribute uint32 Singles;
// end stp class members
};
// end stp class definition

// stp class definition 0
interface CIM_802dot1QVLANService : CIM_VLANService
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 41
interface CAE_VirtualGroupingService : CIM_NetworkService
{
// stp class members
    uint32 create( ());
// end stp class members
};
// end stp class definition

// stp class definition 59
interface CAE_LEC_Service : CAE_ClientService
{
// stp class members
    attribute string ELANName;
    attribute uint16 ELANType;
    attribute uint32 LECID;
    attribute sequence<string> InitialStateParameters;
    attribute string LastError;
    attribute boolean isProxy;
    attribute string PrimaryATMAddress;
    uint32 flushCache(void);
    uint32 join(void);
    uint32 leave(void);
// end stp class members
};
// end stp class definition

// stp class definition 0
interface CAE_VLANService : CAE_VirtualNetworkService
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 76
interface CAE_ELANService : CAE_VirtualNetworkService
{
// stp class members
    attribute uint16 ELANType;
// end stp class members
};
// end stp class definition

// stp class definition 7
interface CIM_ComputerSystem : CIM_System
{
// stp class members
    attribute sequence<string> IdentifyingDescriptions;
    attribute sequence<string> OtherIdentifyingInfo;
    attribute sequence<uint16> Dedicated;
    attribute string NameFormat;
// end stp class members
};
// end stp class definition

// stp class definition 0
interface CAE_LISService : CAE_VirtualNetworkService
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 63
interface CAE_IPOA_ClientService : CAE_ClientService
{
// stp class members
    attribute string IPAddress;
    attribute string PrimaryATMAddress;
    attribute boolean isPVConly;
    attribute string ATMARPServerATMAddress;
    uint32 flushARPCache(void);
// end stp class members
};
// end stp class definition

// stp class definition 40
interface CAE_ATMAdapter : CIM_NetworkAdapter
{
// stp class members
    attribute uint16 AddressType;
    attribute uint16 MaxVPIBits;
    attribute uint16 MaxVCIBits;
    attribute uint16 MaxVPCs;
    attribute uint16 MaxVCCs;
    attribute sequence<string> NetworkAddresses;
    attribute uint16 ConfiguredVPCs;
    attribute uint32 InternalErrors;
    attribute uint16 ConfiguredVCCs;
    attribute uint16 ILMI_VPI;
    attribute uint16 ILMI_VCI;
    attribute uint32 HECErrors;
    attribute uint32 HeaderErrors;
// end stp class members
};
// end stp class definition

// stp class definition 0
interface CAE_LIS : CAE_VirtualNetwork
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 23
interface CIM_LANSegment : CIM_LogicalNetwork
{
// stp class members
    attribute string LANID;
    attribute uint16 LANType;
    attribute string OtherLANType;
```

```
// end stp class members
};
// end stp class definition

// stp class definition 45
interface CAE_IPOAClientEndpoint : CIM_IPProtocolEndpoint
{
// stp class members
// end stp class members
};
// end stp class definition

// stp class definition 28
interface CIM_ServiceServiceDependency
{
// stp class members
    attribute uint16 TypeOfDependency;
// end stp class members
};
// end stp class definition

// stp class definition 29
interface CIM_BindsToLANEndpoint
{
// stp class members
    attribute uint16 FrameType;
// end stp class members
};
// end stp class definition

// stp class definition 31
interface CIM_ActiveConnection
{
// stp class members
    attribute uint16 TrafficType;
    attribute string OtherTrafficDescription;
    attribute boolean isUnidirectional;
// end stp class members
};
// end stp class definition

// stp class definition 30
interface CIM_LogicalNetworkService
{
// stp class members
    attribute uint16 ContainmentType;
// end stp class members
};
// end stp class definition

// stp class definition 37
interface CIM_PhysicalConnector
{
// stp class members
    attribute string ConnectorPinout;
    attribute sequence<uint16> ConnectorType;
// end stp class members
};
// end stp class definition

// stp footer
#endif
// end stp footer
```

Abkürzungsverzeichnis

AMI	ATM management Interface
AAL	ATM Adaption Layer
ATM	Asynchronous Transfer Mode
ATMARP	ATM Address Resolution Protocol
BGP	Border Gateway Protocol
BUS	Broadcast and Unknown Server
CLIP	Classical IP (and ARP over ATM)
CORBA	Common Object Request Broker Architecture
GIOP	General Inter-ORB Protocol
HEC	Header Error Control
IDL	Interface Definition Language
IETF	Internet Engineering Task Force
IIOP	Internet Inter-ORB Protocol
ILMI	Integrated Local Management Interface
IOR	Interoperable Object Reference
IP	Internet Protocol
IPOA	IP over ATM
LAN	Local Area Network
LANE	LAN Emulation
LEC	LAN Emulation Client
LECS	LAN Emulation Configuration Server
LES	LAN Emulation Server
LIS	Logical IP Subnet
MAC	Medium Access Control
MASA	Mobile Agent System Architecture
MIB	Management Information Base
MTU	Maximum Transmission Unit
NNI	Network-to-Network Interface
OAM	Operation and Maintenance
OMA	Object Management Architecture
OMG	Object Management Group
OMT	Object Modelling Technique
ORB	Object Request Broker
PHY	Physical Layer (Bitübertragungsschicht)
PVC	Permanent Virtual Connection
QoS	Quality of Service
RFC	Request for Comment

RMON	Remote Networking Monitoring MIB
SNMP	Simple Network Management Protocol
SPANS	Simple Protocol for ATM Network Signalling
SVC	Switched Virtual Connection
StP	Software through Pictures
TCP	Transmission Control Protocol
UNI	User-to-Network Interface
URL	Uniform Resource Locator
VCC	Virtual Channel Connection
VCI	Virtual Channel Identifier
VCL	Virtual Channel Link
VPC	Virtual Path Connection
VPI	Virtual Path Identifier
VPL	Virtual Path Link
WWW	World Wide Web

Literaturverzeichnis

[AhTe 94] AHMED, M. und K. TESINK: *RFC 1695: Definitions of Managed Objects for ATM Management Version 8.0 using SMIv2*. RFC, IETF, August 1994.

[BHK 95] A. BADACH, E. HOFFMANN und O. KNAUER: *High Speed Internetworking: Grundlagen und Konzepte für den Einsatz von FDDI und ATM*. Addison-Wesley, 1995.

[Black 95] BLACK, U.: *ATM Volume I: Foundation for Broadband Networks*. Prentice Hall, 1995.

[Black 98a] BLACK, U.: *ATM Volume II: Signaling in Broadband Networks*. Prentice Hall, 1998.

[Black 98b] BLACK, U.: *ATM Volume III: Internetworking with ATM*. Prentice Hall, 1998.

[CIM 98a] DISTRIBUTED MANAGEMENT TASK FORCE: *Common Information Model (CIM) Specification Version 2.0*, 1998.
Verfügbar via WWW http://www.dmtf.org/spec/cim_spec_v20.

[CIM 98b] DISTRIBUTED MANAGEMENT TASK FORCE: *CIM Specification Errata Sheet v2.0.2*, 1998.
Verfügbar via WWW http://www.dmtf.org/spec/cim_errata.html.

[CIM 98d] DISTRIBUTED MANAGEMENT TASK FORCE: *CIM Core Model White Paper Version 1.0*, 1998.
Verfügbar via WWW http://www.dmtf.org/spec/cim_core.

[CIM 99a] DISTRIBUTED MANAGEMENT TASK FORCE: *Common Information Model (CIM) Specification Version 2.2*, 1999. Verfügbar via WWW http://www.dmtf.org/download/spec/cim_spec_v22.pdf.

[CIM 99b] DISTRIBUTED MANAGEMENT TASK FORCE: *CIM Schema Version 2.2*, 1999.
Verfügbar via WWW http://www.dmtf.org/spec/cim_schema_v22.html.

[Date 95] DATE, C. J.: *An Introduction to Database Systems*. Addison-Wesley, 1995.

[Fore 96a] FORE SYSTEMS, INC.: *ForeRunner ATM Switch Configuration Manual*, 1996.

[Fore 96b] FORE SYSTEMS, INC.: *ForeRunner HPA-200 ATM EISA Bus Adapter User's Manual*, 1996.

[GLWK 98] GREENE, M., J. LUCIANI, K. WHITE und T. KUO: *RFC 2320: Definitions of Managed Objects for Classical IP and ARP Over ATM Using SMIv2 (IPOA-MIB).* RFC, IETF, April 1998.

[Guts 95] GUTSCHMIDT, M.: *Ein Objektmodell für ein integriertes Management von Systemdiensten mit Client/Server-Struktur.* Dissertation, Ludwig-Maximilians-Universität München, September 1995.

[HAN 99] HEGERING, H.-G., S. ABECK und B. NEUMAIR: *Integrated Management of Networked Systems – Concepts, Architectures and their Operational Application.* Morgan Kaufmann Publishers, 1999.

[Hein 97] HEIN, M. und N. VON DER LANCKEN: *ATM, Konzepte–Trends–Migration.* International Thomson Publishing, 1997.

[IBM 98] IBM, INC.: *ATM MIF proposal*, 1998.

[IEEE 98] IEEE-802.1-WG: *Draft Standard P802.1Q/D11 - IEEE Standards for Local and Met ropolitan Area Networks: Virtual Bridged Local Area Networks.* Technischer Bericht IEEE, Juli 1998.

[ILMI 96] *Integrated Local Management Interface (ILMI) Specification, Version 4.0.* Approved Specification af-ilmi-0065.000, ATM Forum, September 1996.

[Kemp 98] KEMPTER, B.: *Entwurf eines Java/CORBA-basierten Mobilen Agenten.* Diplomarbeit, Technische Universität München, August 1998.

[Kyas 96] KYAS, O.: *ATM-Netzwerke, Aufbau–Funktion–Performance.* DATACOM Buchverlag, 1996.

[LaHa 98] LAUBACH, M. und J. HALPERN: *RFC 2225: Classical IP and ARP over ATM.* RFC, IETF, April 1998.

[LANE 97] *LAN Emulation over ATM, Version 2 — LUNI Specification.* Approved Specification af-lane-0084.000, ATM Forum, Juli 1997.

[Laub 94] LAUBACH, M.: *RFC 1577: Classical IP and ARP over ATM.* RFC, IETF, Januar 1994.

[LECM 97] *LAN Emulation Client Management Specification, Version 2.0.* Approved Specification af-lane-0093.000, ATM Forum, Oktober 1998.

[LESM 96] *LAN Emulation Servers Management Specification, Version 1.0.* Approved Specification af-lane-0084.000, ATM Forum, März 1996.

[M4NM 99] *M4 Interface Requirements and Logical MIB: ATM Network View, Version 2.* Approved Specification af-nm-0058.001, ATM Forum, Mai 1999.

[MASA 99] GRUSCHKE, B., S. HEILBRONNER und H. REISER: *Die Mobile Agent System Architecture MASA – Eine Java- und Corba-basierte Ablaufumgebung für integriertes Management.* Technischer Bericht Ludwig-Maximilians-Universität München, Institut für Informatik, 1999.

[McRo 91] McCloghrie, K. und M. T. Rose: *RFC 1213: Management Information Base for Network Management of TCP/IP-based internets:MIB-II.* RFC, IETF, März 1991.

[MPOA 98] *Multiprotocol Over ATM, Version 1.0 MIB.* Approved Specification af-mpoa-0092.000, ATM Forum, Juli 1998.

[MPOA 99] *Multiprotocol Over ATM Specification, Version 1.1.* Approved Specification af-lane-0114.000, ATM Forum, Mai 1999.

[Muel 98] Müller, T.: *CORBA-basiertes Management von UNIX-Workstations mit Hilfe von ODP-Konzepten.* Diplomarbeit, Technische Universität München, Februar 1998.

[NST 99] Noto, M., E. Spiegel und K. Tesink: *RFC 2514: Definitions of Textual Conventions and OBJECT-IDENTITIES for ATM Management.* RFC, IETF, Februar 1999.

[Oest 98] Oestereich, B.: *Objektorientierte Softwareentwicklung – Analyse und Design mit der Unified Modeling Language.* Oldenbourg Verlag, 1998.

[Pry 94] de Prycker, M.: *Asynchronous Transfer Mode: Die Lösung für Breitband-ISDN.* Prentice Hall, 1994.

[RJB 98] Rumbaugh, J., I. Jacobson und G. Booch: *Unified Modeling Language — Reference Manual.* Addison-Wesley, 1998.

[RMON 97] *ATM Remote Monitoring SNMP MIB.* Approved Specification af-nm-test-0080.000, ATM Forum, Mai 1997.

[Roel 99] Roelle, H.: *Prototypische Implementierung eines CORBA Topology Service.* Fortgeschrittenenpraktikum, Technische Universität München, Februar 1999.

[Sche 99] Schenkl, M.: *Entwicklung von Praktikumsaufgaben auf Basis des ATM-Testbeds für das Rechnernetze-Praktikum.* Diplomarbeit, Technische Universität München, Mai 1999.

[StP 96a] Interactive Development Environments, Inc.: *Software through Pictures User Manual: StP Core – Fundamentals of StP*, 1996.

[StP 96b] Interactive Development Environments, Inc.: *Software through Pictures User Manual: Creating Models*, 1996.

[StP 96c] Interactive Development Environments, Inc.: *Software through Pictures User Manual: Generating Code*, 1996.

[Tan 96] Tanenbaum, A. S.: *Computer Networks.* Prentice Hall, 1996.

[TeEd 99] Tesink, K. und Ed: *RFC 2515: Definitions of Managed Objects for ATM Management.* RFC, IETF, Februar 1999.

[Wald 91] Waldbusser, S.: *RFC 1271: Remote Network Monitoring Management Information Base.* RFC, IETF, November 1991.

[WBEM 98] Microsoft Corporation: *Web-Based Enterprise Management (WBEM) Home Page*, 1998.
Verfügbar via WWW http://www.microsoft.com/management/WBEM.