Service and Network Management in the OAMS[1] Open Service Architecture

Simon ZNATY
Swiss Federal Institute of Technology
TCOM Laboratory
Telecommunications Services Group
CH-1015 Lausanne, Switzerland
E-mail : znaty@tcom.epfl.ch

Abstract

New open service architectures are now emerging to ease service construction, management, testing, deployment, and operation, and to hide from the service designer the heterogeneity of the underlying technologies and the complexities introduced by distribution. These architectures should provide all the functionalities for call and connection management, and support management functional areas. To reach these objectives, information models must be defined which provide service and network representations by data. This paper presents the specification, design and implementation of the management architecture of OAMS which is an open service architecture. The proposed architecture is deployed over the distributed processing environment DCE[2]. It exhibits service, network and element management views by means of information models, in order to manage the end-to-end connectivity requested by telecommunication services.

I. Introduction

The demand for new sophisticated services such as multimedia services is on the increase. Short term Intelligent Network (IN) approaches [1] are addressing call related services and customization of such services. However, multimedia services are more than simple extensions to basic calls. They require more management, more flexible connectivity, better charging and security than the current IN is capable of providing.

Therefore we have defined a software architecture named OAMS [2] which objectives are the introduction and the management of advanced telecommunication services or multimedia services and the management of the underlying telecommunication networks that are used to provide those services. OAMS supports distributed telecommunications applications, connection management, and service and network management.

Although our proposed architecture is generic in regard to the underlying telecommunication network technology, it is described with an end-to-end ATM based communication platform. ATM (Asynchronous Transfer Mode) [3] is the advanced multiplexing and switching technique, chosen as the target transfer mode solution for implementing Broadband ISDN.

OAMS consists of four levels, telecommunication service architecture, service and network management architecture, distributed computing environment and communication platform with a major role given to software (Figure 1). Indeed OAMS embraces TMN (Telecommunication Management Network) [4] within a framework based on distributed processing (ODP) principles [5] and object orientation. The ODP architecture resolves

---

[1] OAMS stands for **O**pen Service **A**rchitecture for **M**ultimedia **S**ervices over ATM
[2] DCE stands for Distributed Computing Environment

integration problems of distribution, interworking and portability and provides a general standardized framework for the design of distributed information processing systems. Object orientation is the core technology used for service creation and management and network management because it promotes high reusability, extendibility and maintainability that are current requirements on telecommunication software modules. The management architecture of OAMS extends the ITU-T element management model [6]. This latter has been employed in managing switching elements but will not be sufficient for the new task of service management. This latter task requires management interfaces that hide the underlying complexity of networks.

OAMS objectives are similar to TINA [7], ROSA [8] and CASSIOPEIA [9] objectives (i.e., develop an Open Service Architecture), but the concepts from which OAMS is built are different from those of these latter projects, particularly with regard to service introduction with the definition of a methodological support to build telecommunication services [10], and service management with the provision of a framework to manage multimedia and host resources and user applications [11].

In this paper, we present the service and network management architecture of OAMS which major components are service and network management information models. These latter provide service and network representations by data respectively. They contain common object classes used for the development of management functional areas (e.g., connection, performance, fault management, etc). Three main managed object classes are used to describe all models. These are node, link and access point.

The OAMS network management architecture provides the management of the end-to-end multi-network to which users access, of subnetworks that constitute it, and of the network elements within these subnetworks. Most standard object classes defined in ITU-T M.3100 [6] appear in the lowest level of the OAMS management architecture since the main focus of this recommendation is on network element aspects. The proposed architecture is applicable to both private (Customer Private Network) and public (Telecommunications Network) domains.
Then, the proposed management information models are encapsulated into computational objects and implemented with DCE which is the distributed processing environment that supports the distributed execution of telecommunications and management applications deployed over the OAMS architecture.
Finally a connection management function is designed and implemented, relying on those information models. This function enables establishing, releasing and managing connections with long holding times (i.e., permanent or semi-permanent connections). Setting up connections that have short holding times is the purpose of signalling [12].
In [13] an open architecture called XBIND is proposed which aims at supporting a multitude of applications such as connection management and distributed computing. However this connection management architecture is signalling-based while in OAMS it is network management-based and therefore only concerns long term connections.

The paper is organized as follows. The second section introduces the management architecture of OAMS. The first subsection details the several management layers present in this architecture and the relationships among these layers. The second subsection describes the information models proposed at every management layer. The third subsection details how to pass from information models to information managers, i.e., how to derive a functional architecture from an information architecture. Those information managers provide the basic functionalities required for the development of system management functional areas.
The third section details the connection management function.
The fourth section deals with the deployment of the OAMS management architecture on a network of HP workstations interconnected by a Fore ASX 200 ATM switch. The tools used

are DCE for distribution transparency, mSQL for data storage, and CMU-SNMP for SNMP capabilities. These tools are also briefly described.
In the conclusion, we present the integration of our results into OAMS, in particular how telecommunications services can rely on the connection management function.

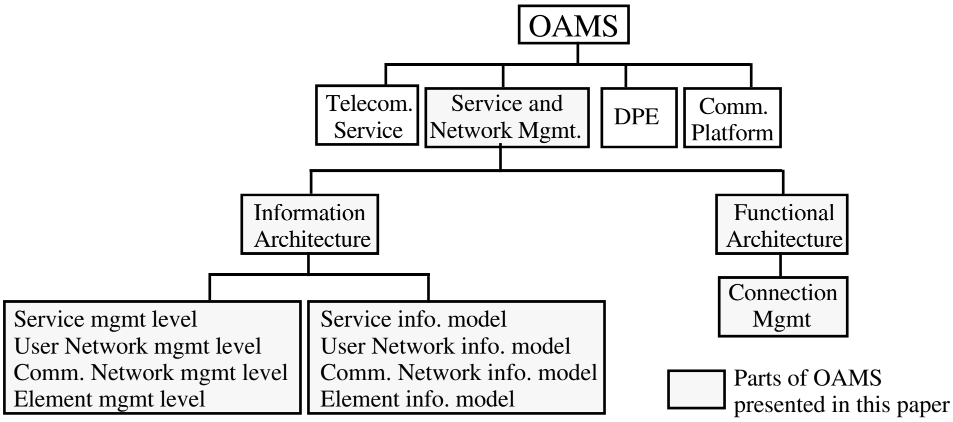


Figure 1: OAMS overall view and its components presented in this paper

II. Service and Network management architecture of OAMS

The service and network management architecture of OAMS consists of two kinds of architectures: an information architecture and a functional architecture. The information architecture is made of a set of management layers and information models present at every management layer.
The functional architecture shows a set of functional objects that interact with each other through interfaces. These functional objects encapsulate the information models and implement the service or network management functionalities. In the following subsections, emphasis will be put on the management layers considered in the context of OAMS, on the information models at these management layers, and on the functional objects.

II.1. The considered management layers

The notion of visibility levels is very useful for defining the management layers where an information model should be present in order to provide information for the purpose of management. A visibility level represents the level of abstraction at which the system is considered. It contains the information necessary to take decisions. The management layers defined within TMN are examples of such visibility levels. Our information structuring makes use of these management layers and subdivides them when necessary for more accurate management. Therefore, several visibility levels may be defined within a given management layer. The TMN layers are the service, network and element management layers.

• The service management layer (SML) is concerned with the contractual aspects of services that are provided to customers. This layer provides the customer interface and interacts with the network management layer.

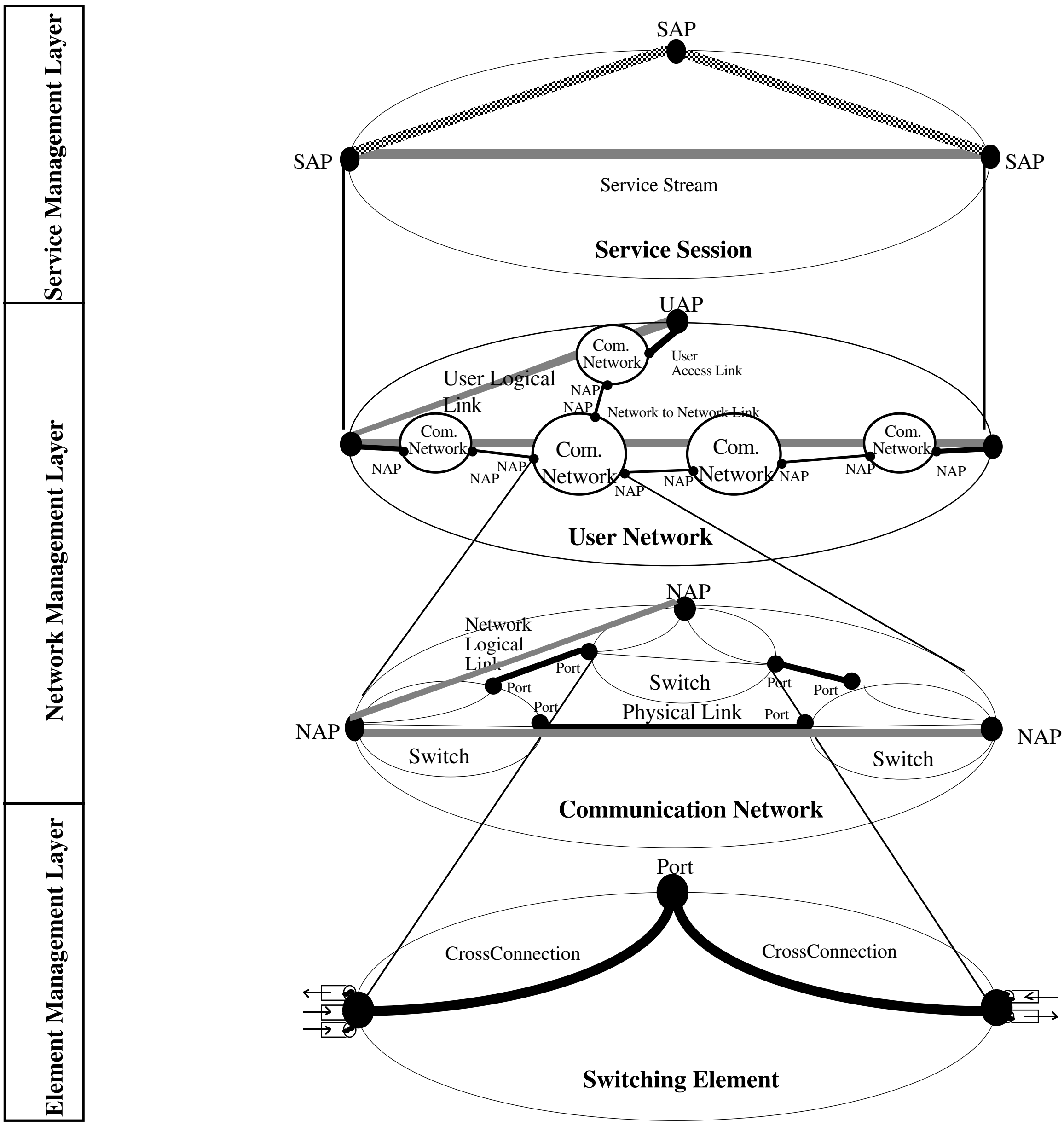
• The network management layer (NML) covers two visibility levels, presented hereafter:
  • The user network management level (UNML) provides an end-to-end view of the telecommunications resources involved to provide applications with the communications capabilities they need. This level encompasses both private and public domains and acts as a server to the service management layer.
  • The communication network management level (CNML) reveals the internal structure of the public or private network using the partitioning rule. It is in charge of controlling and

coordinating the communication network view of network elements. Note that a communication network can itself be composed of lower communication networks.

• The element management layer (EML) describes the connectivity offered by individual switching elements.

The service and user network management views are maintained by the service provider while the communication network and element management levels are updated by network operators.

Figure 2 illustrates how TMN management layers and visibility levels are interrelated in the context of the OAMS management architecture.



SAP : Service Access Point **UAP** : User Access Point **NAP** : Network Access Point

Figure 2: The Visibility levels within the management architecture of OAMS

II.2. The service and network management information models

This section provides detailed explanations on the different levels of the OAMS service and network management information architecture and their associated management information models. The design of these models follows the OSI guidelines [14] and thus is object oriented. At every management level, these models consist of object classes and relationships among these classes.

The OMT[3] notation [15] is used to graphically represent the proposed information models while GDMO[4] [16] and GRM[5] [17] textual notations are used to specify these models semi-formally. The GDMO standard provides a set of templates to specify managed object classes. The GRM standard provides, in the form of templates, general and flexible relationship modelling mechanisms to specify n-ary relationships among managed objects.

Information models here represent service or network resources by management information. The amount of information that must be managed requires that a *structured* approach be taken to represent that information. Since many service and network entities share common characteristics, a way needs to be found to exploit this characteristic in order to maximize *commonality* and minimize implementation efforts and diversity. "Structured" calls for a model. "Commonality" requires a model that is applicable to all service or network entities. Therefore, *generic* service and network management information *models* are the means to have common models.

The models at the {service, user network, communication network, switching element} management levels are generic, therefore topology and technology independent. They can be applied to private or public domains. They consist of three main entities, namely node, link and access point, and of generic relationships among these entities.

The following subsections present the proposed information models.

II.2.1. The information model at the service management level

The service management layer provides a level of abstraction where the topology and technology of the underlying networks are transparent, that is, the information model developed at this level is independent of these aspects. Therefore its main focus is on the modelling, monitoring, control and management of flows of information between service access points (Figure 2). Concepts useful for that purpose have been borrowed from [18][19][20].

A service access point is defined as the point of interaction between a service and a service provider that may be identified with an address.

A service component is a service object providing the communication capabilities for a single information type. Generic information types defined so far are audio, video, text, data, pictures and graphics. Quality classes have been defined for each information type. They allow the specification of the standards and coding as well as the throughput to characterize the type of information transferred.

A service task is that part of a telecommunication service which deals with the communication related aspects rather than the administrative aspects or those related to information type. A service task comprises one or several service components as well as information related to the

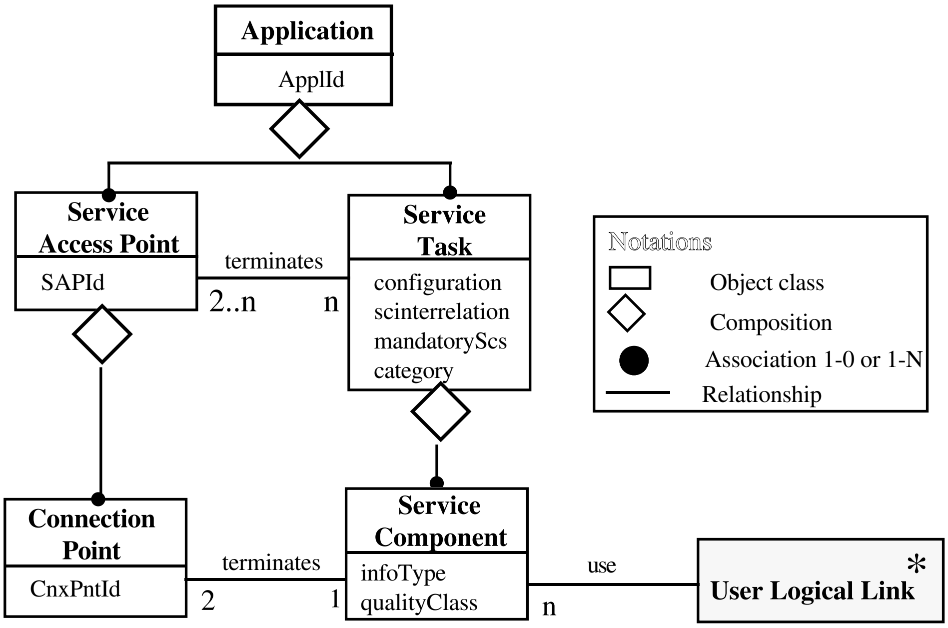
---

[3] Object Modeling Technique

[4] Guidelines for the Definition of Managed Objects

[5] General Relationship Model

configuration (point-to-point, point-to-multipoint, etc.), and symmetry (unidirectional, bi-directional, etc.) of the overall information transfer. In addition, interrelations between service components may be specified at the service task level (for instance if synchronization between audio and video streams is required).
The concept of service task is quite similar to well-known concepts such as "call" or "session" but with a special emphasis on the communications issues, that is, the information flows and groupings thereof, rather than on the users involved. The obtained model is represented in Figure 3 using the OMT notations.



(* : Class Described at the User Network Management Level)

Figure 3: The information model at the service management level

II.2.2. The information model at the user network management level

The information model at the user network management level (UNML) represents the global network which is used by the user (Figure 2). This latter accesses through a particular point, called UAP (User Access Point). The user communications are modelled as User Logical Links (ULLs) within the global network. This logical link that connects two UAPs uses the User Access physical Link (UAL) from the UAP to the Network Access Point (NAP) to reach the first communication network to which the user is connected. It crosses this communication network by means of a network logical link (modelled at the communication network management level). If several communication networks have to be crossed to enable communications between two UAPs, passing from one communication network to another is via a Network to Network Link (NNL). The obtained model is represented in Figure 4. Note that it is independent of any network technology.

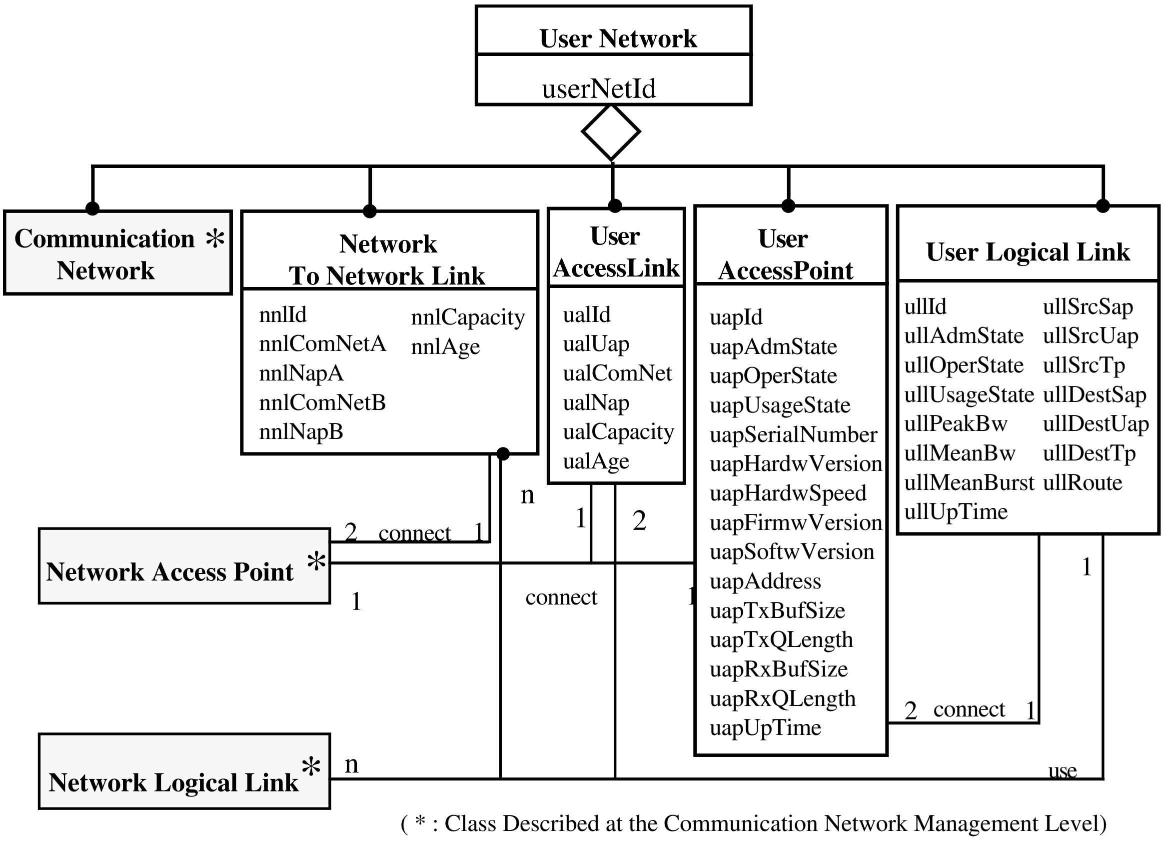


Figure 4: The information model at the user network management level

II.2.3. The information model at the communication network management level

The information model defined at the communication network management level (CNML) represents an information transport network which is public or private (Figure 2). It consists of switching elements which are the network nodes, of physical links between these nodes, and of ports which terminate these links at the switches' level. The communication network can be accessed through particular points, called NAPs (Network Access Points). A communication between two NAPs within the communication network is modelled as a network logical link (NLL). A network logical link is an association of cross-connections and physical links, and provides the end-to-end connectivity within a communication network, used by the upper management levels.
The communication network management information model contains the managed switching elements present at the element management level since it has the responsibility for their management. Therefore, interactions take place between these two management levels. This model is derived from the one proposed in [21]. The obtained model is represented in Figure 5.

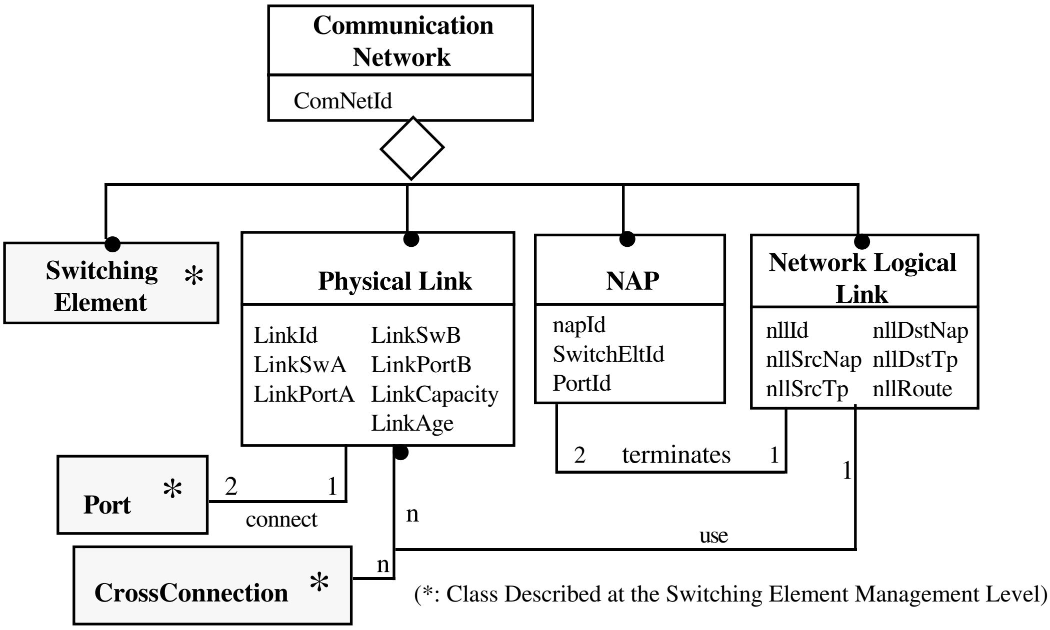


Figure 5: The information model at the communication network management level

II.2.4. The information model at the switching element management level

The element management information model represents the connectivity offered by individual switching elements with ports and cross-connections between them (Figure 2). Classes "Switching element" and "Port" may be related to "Fabric" and "TpPool" managed objects defined in ITU-T M3100. A Fabric represents the function of managing the establishment and release of cross-connections. It also manages the assignment of termination points to TPPools. TPPool identifies a collection of termination points and maintains their availability status. It is used for some management purpose such as routing. The class "CrossConnection" is similar to the "CrossConnection" managed object also defined in ITU-T M3100. In our model, only one kind of connection termination point (CTP) has been considered whereas [6] distinguishes between source, sink and bi-directional CTPs.
The only quality of service information associated with Ports are *availability* which is expressed by a state associated with each of these elements, and *throughput* (maximum bandwidth, allocated bandwidth and used bandwidth). The obtained model which is network technology independent is represented in Figure 6.

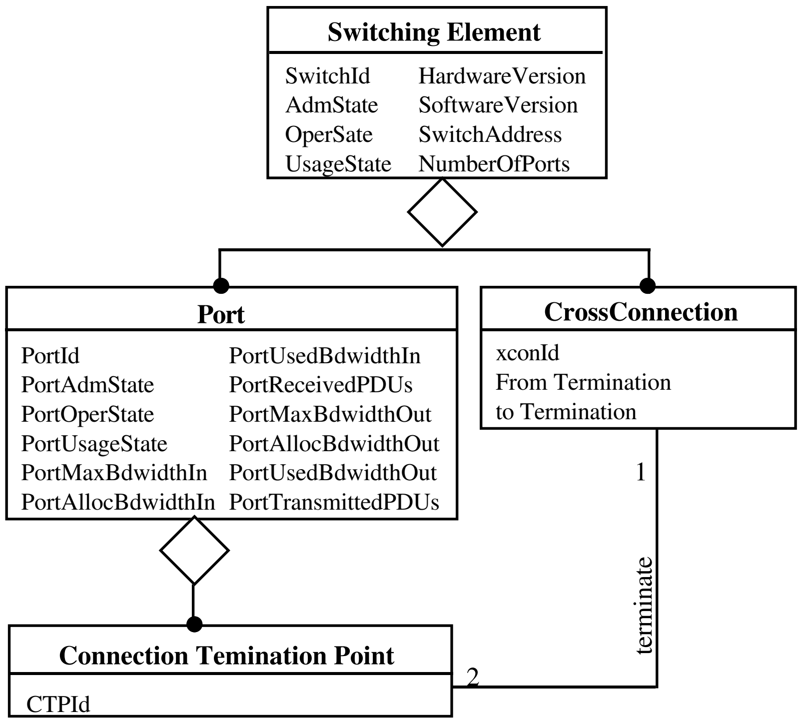


Figure 6: The information model at the switching element management level

Figure 1 is only illustrative and not prescriptive regarding the configuration it represents (e.g., number of communication networks, of switching elements, etc). The corresponding object models (Figure 3 to Figure 6) are general enough to represent any relevant configuration. They all have a similar generic structure.

II.3. From Information Models to Information Managers

At every visibility level, information models are translated into computational objects (COs). These COs encapsulate and maintain the management view of the services, networks and network elements; they are called *information managers*. According to ODP [5] which provides a reference model for building open distributed systems, a CO provides a set of services that can be used by other COs. To enable other COs to access its services, a CO offers computational interfaces, which are the only means by which other objects can use the services. Therefore, COs interact with each other through interfaces that they provide. The interfaces are of operational type. An operational interface enables interactions that are operations and possibly, responses to these operations. Figure 7 illustrates the interactions between information managers and COs modelling management functional areas (called *functional managers*). Note that UNML, CNML and EML information managers provide a CMIS (Common Management Information Service) interface while the agents of ATM switches at the network element layer (NEL) supply an SNMP interface.

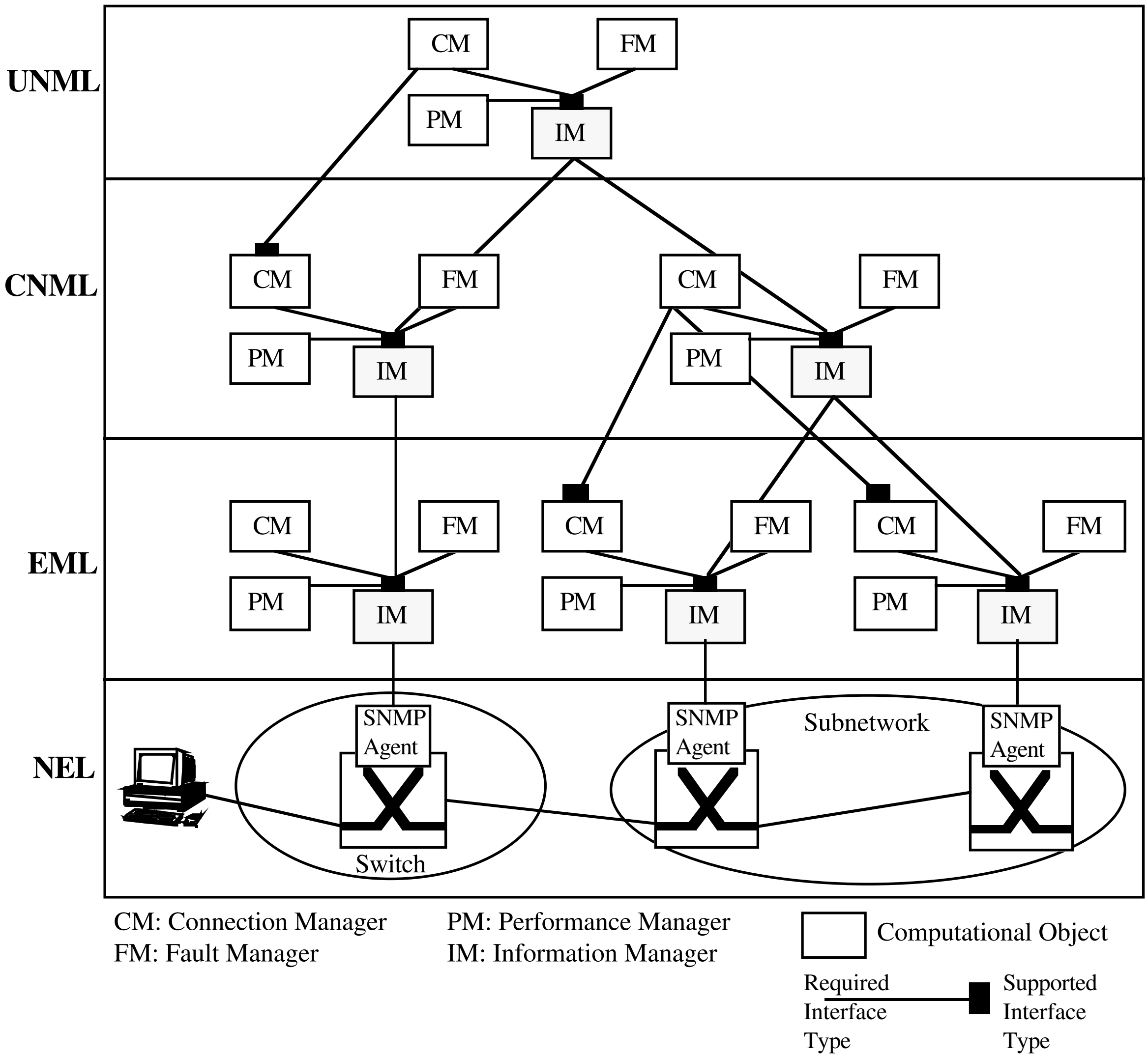


Figure 7: Interactions between functional managers and information managers

III. Connection management in OAMS

In this section, emphasis is put on the connection management function with its connection managers (i.e., functional managers implementing connection management) developed within the OAMS management architecture. It enables establishing, controlling and releasing connections with long holding times (i.e., permanent or semi-permanent connections) requested by telecommunications services such as broadband virtual private networks (BVPN). For example, it allows a customer modifying the configuration of his BVPN, by adding new sites and creating and removing virtual private lines (semi-permanent connections) between them.

The different actors taking place in the connection management architecture are the user requesting a service, the service provider offering service sessions to users, and the network provider whose responsibility is to provide connections .
For the support of the connection management function, the management architecture of the service provider consists of a service session manager and a user network connection manager.

The service session manager provides the functionalities needed to create, delete and modify service sessions. A service session is the context supporting the activities and the execution of a service.
The user network connection manager provides the functionalities needed to handle and control the communication resources. It controls the connectivity and the allocation of resources related to the session connections. A session connection is the context supporting the activities and functions for establishing connections related to a service session between users. A service session is oriented towards users while session connections are oriented towards user terminals.
For the support of the connection management function, the management architecture of the network provider consists of a communication network connection manager and a set of element connection managers. If several administrative domains have to be crossed to connect the users, several communication network connection managers may be present, one for each administrative domain.
Figure 8 provides an overall view of the connection managers (CMs) present at every management level and interacting with the information managers (IMs) of the corresponding level. A connection manager is responsible of establishing, releasing and managing the connections at a given level.

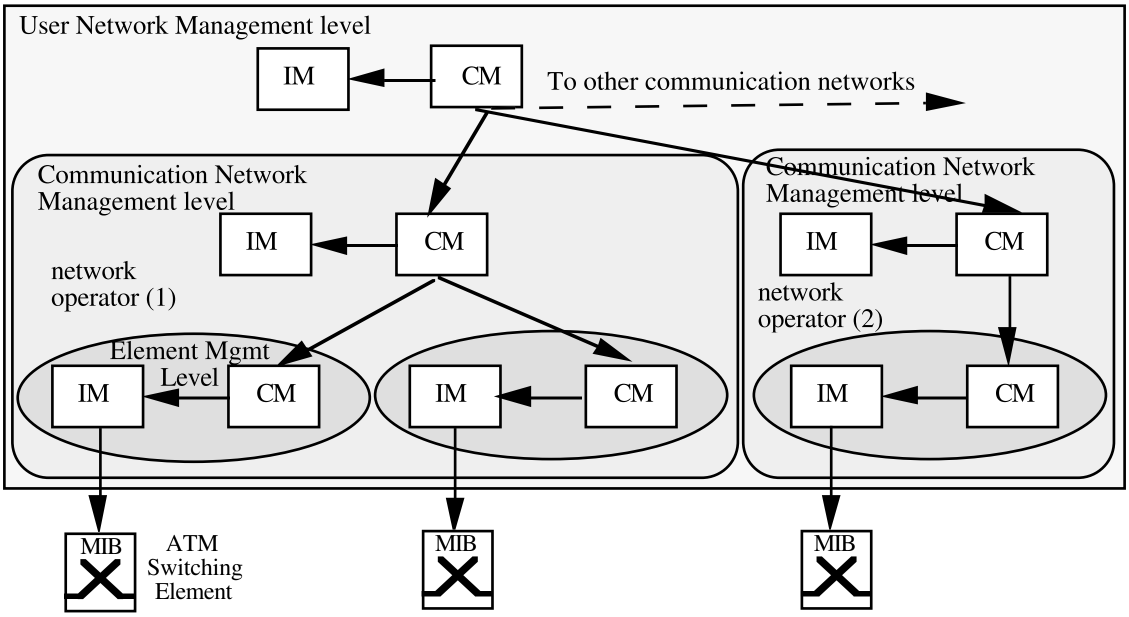


Figure 8: overall view of the connection management function

III.1. Connection establishment

A connection manager at the UNML or CNML relies on the services supplied by a routing process to establish a connection (Figure 9). The router computational object (CO) is responsible to select the best-suited route for the required connection. At the UNML level, the router selects the route for establishing a user logical link. Its view is only that of communication networks and links among them. At the CNML level, the router finds the best-suited path for establishing a network logical link within a given communication network.

Based on the information provided by the router, the connection manager creates at the UNML (resp. CNML) level the different links composing the route and allocates to each segment its associated bandwidth, link number and Quality of service (QoS) parameters. At the EML level, cross-connections are performed on every switching element on the route.

Figure 9 details the interactions among the connection managers, information managers and routers at the UNML, CNML and EML management levels.

The operations associated to these interactions are summarized in tables 1, 2 and 3.

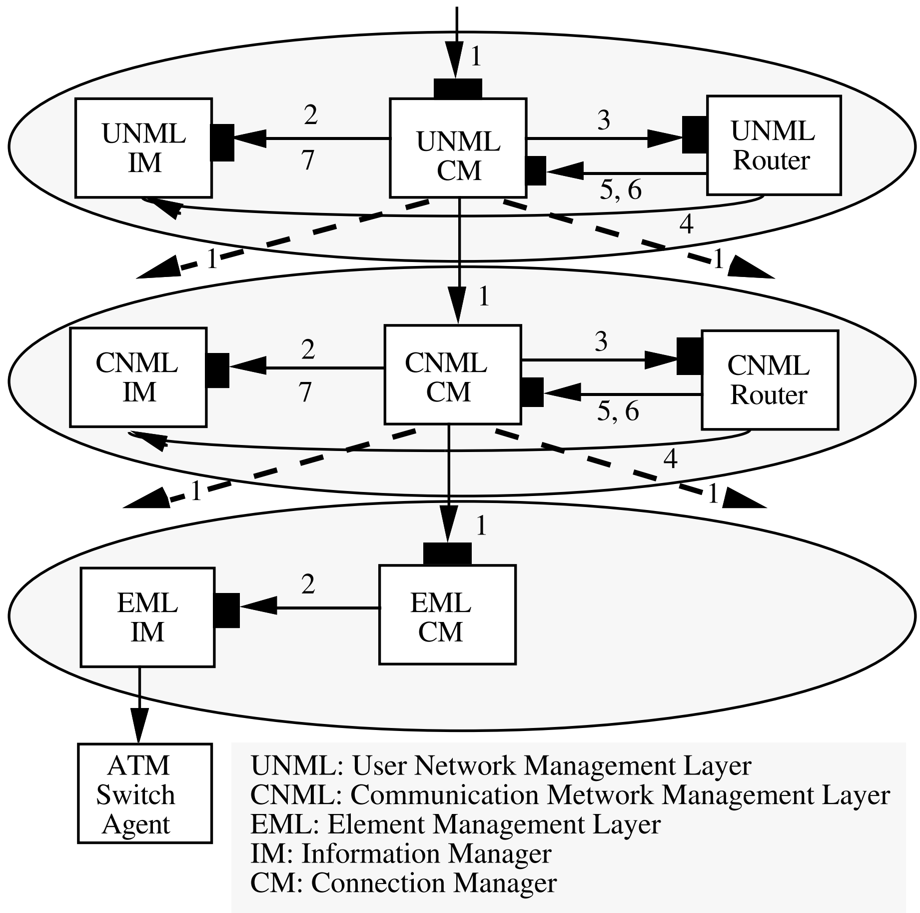


Figure 9: Connection establishment in OAMS

Nb	Operation	Description
1	ullSetup()	Requests the establishment of a UserLogicalLink (ull)
2	ullCreate()	Requests the creation of a new UserLogicalLink object in the UNML IM
3	ullRoute()	Requests to the UNML router the route to setup the ull
4	unmlGet()	Reads information from the UNML IM
5	ullnllInit()	Establishes a first NetworkLogicalLink (nll)
6	ullnllExpand()	Establishes each subsequent nll
7	ullRouteSet()	Updates the ullRoute field in the UNML IM

Table 1: Operations offered by the connection mgr, information mgr and router at the UNML level

Nb	Operation	Description
1	nllSetup()	Requests the establishment of a NetworkLogicalLink (nll)
2	nllCreate()	Requests the creation of a new NetworkLogicalLink object in the CNML IM
3	nllRoute()	Requests to the CNML router the route to setup the nll
4	cnmlGet()	Requests information from the CNML IM
5	nllxconInit()	Establishes a first cross-connection
6	nllxconExpand()	Establishes each subsequent cross-connection
7	nllRouteSet()	Updates the nllRoute field in the CNML IM

Table 2: Operations offered by the connection mgr, information mgr and router at the CNML level

Nb	Operation	Description
1	XconSeup()	Requests the set-up of an CrossConnection (Xcon)
2	emlconCreate()	Creates a new CrossConnection in EML IM

Table 3: Operations offered by the connection manager and information manager at the EML

III.2. Connection release and control

In a similar way, the connection managers are also responsible of releasing and controlling connections. These procedures are performed through interactions between connection managers and information managers. A connection release consists in deleting in the UNML, CNML and EML information models the corresponding rows in the ULL, NLL and XCON tables.
The control of a connection consists in modifying the QoS parameters associated with a given connection at the UNML, CNML and EML levels.

IV. Implementation of the OAMS management architecture

In this section, we describe the implementation of the information managers in the distributed computing environment (DCE) version 1.0 [22]. DCE is a set of software tools and services from Open Software Foundation (OSF) for developing and running distributed applications in heterogeneous environments. DCE internally works with the client/server model and is particularly well-suited for the development of applications that are structured according to this model: a server typically offers some service to a population of clients.
At every management level, information models encapsulated within information managers are maintained by these latter. The information models designed using the object oriented approach are translated into SQL tables stored in the relational database management system mSQL [23]. mSQL is a lightweight database engine which provides a subset of ANSI SQL as its query interface. Both object classes and relationships among these classes are mapped to SQL tables.
Figure 10 shows how to pass from a Port object class to a Port SQL table. In this case, one class converts to one table, but this may be not systematically the case. The class Port has among its attributes PortId. The Table model lists these attributes and adds the implicit object ID such as SwitchId. As part of formulating the table model, we add details. We specify that SwitchId and PortId cannot be null since they are candidate keys, etc. The SQL code creates the Port table. This method is derived from the one proposed in [15].

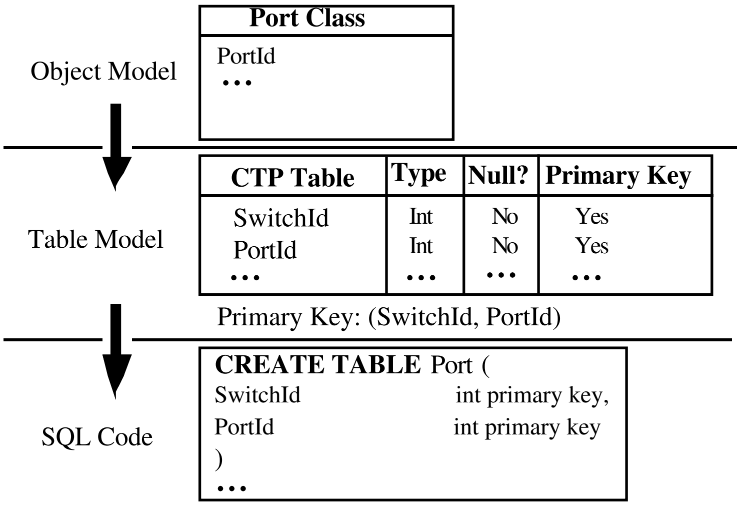


Figure 10: From object classes to SQL tables

Hereafter, we describe the operation of an EML information manager developed on DCE. The CO providing the abstract and structured view of the switching element at the EML level (i.e., EML information manager) has been implemented with C over DCE. It consists of two parts, namely service user and service provider.

• The *service user* entity updates the objects of the information model at the EML level, encapsulated in this CO. This update is performed thanks to data contained in the FORE SNMP MIB of the ATM switch. These data are related to Port and cross-connection objects that constitute the model. The management requests sent to the ATM switch agent are implemented with the API of the CMU-SNMP software package that provides SNMPv1 and SNMPv2 services. We make use of the former services. These services are Snmpget, Snmpget-next, Snmpset, Snmpwalk. Replies returned to the EML information manager are stored in a relational database which implements the information model at the EML level. Figure 11 illustrates the relationships among management layers.

• The *service provider* entity handles requests emitted by the NML information manager. Since CNML requests are performed using CMIS, the EML information manager implements a CMIP-to-SQL translator for M-get operation, a CMIP-to-SNMP translator for M-set operation since this latter should modify the behavior of real resources. Note also that the SNMP agent of the Fore ATM Switch allows creating and deleting rows in SNMP tables (e.g., creation or release of a new connection in a connection table) using the SNMP-set operation.

Figure 10 presents the implementation architecture relying on the DCE environment. Each computational object is mapped into a DCE process and the operations offered by this computational object are grouped into an interface. Information managers offer a CMIS interface and connection managers supply interfaces described in tables 1, 2 and 3. These interfaces include high level operations that rely on CMIS.

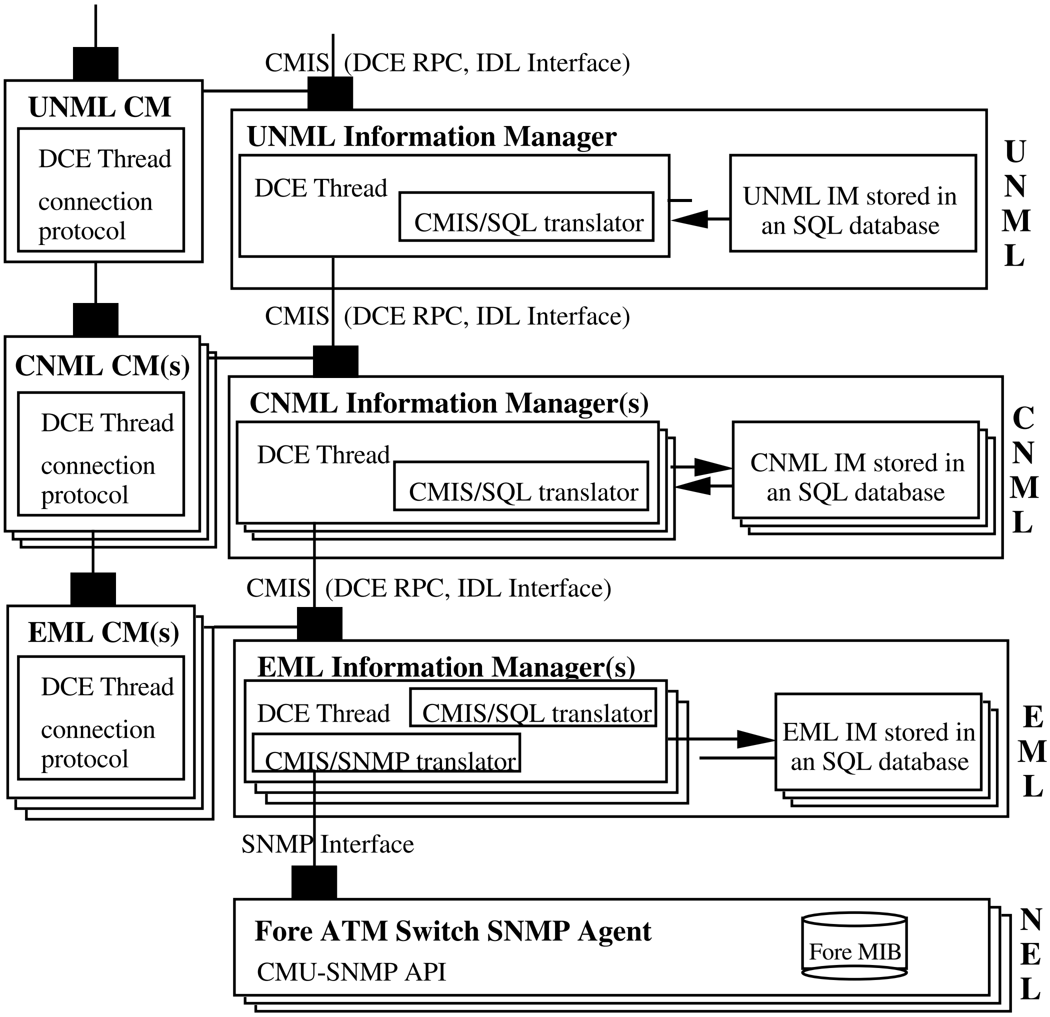


Figure 11: Relationships and translations among management layers

With OSF DCE version 1.0, each generated process is nearly 2.5 Mbytes large. Efficiency and memory constrains is not yet a real problem since few processes are present, i.e., a UNML information manager, a UNML connection manager, a CNML information manager, a CNML connection manager, N element managers (one for each ATM Switch) and N element connection managers. Problems will occur when other functional managers will be added (e.g., performance managers). In this case, each set of functional managers and their associated information manager will be mapped to one DCE process all together, in order to minimize their number. We have also to take into consideration that DCE version 1.1 which exists commercially and represents a major advance over version 1.0 should help in solving the problem of memory and efficiency.

To validate the connection management architecture, we have developed using the methodology proposed in [10] a broadband virtual private network (BVPN) service with its associated configuration management system (Figure 12). This latter enables a BVPN customer dynamically modifying his BVPN configuration, i.e., adding or removing logical links (ULLs) between the different sites of his organization. Currently, establishing an ULL with the proposed architecture considering one ATM switch takes 4 seconds.

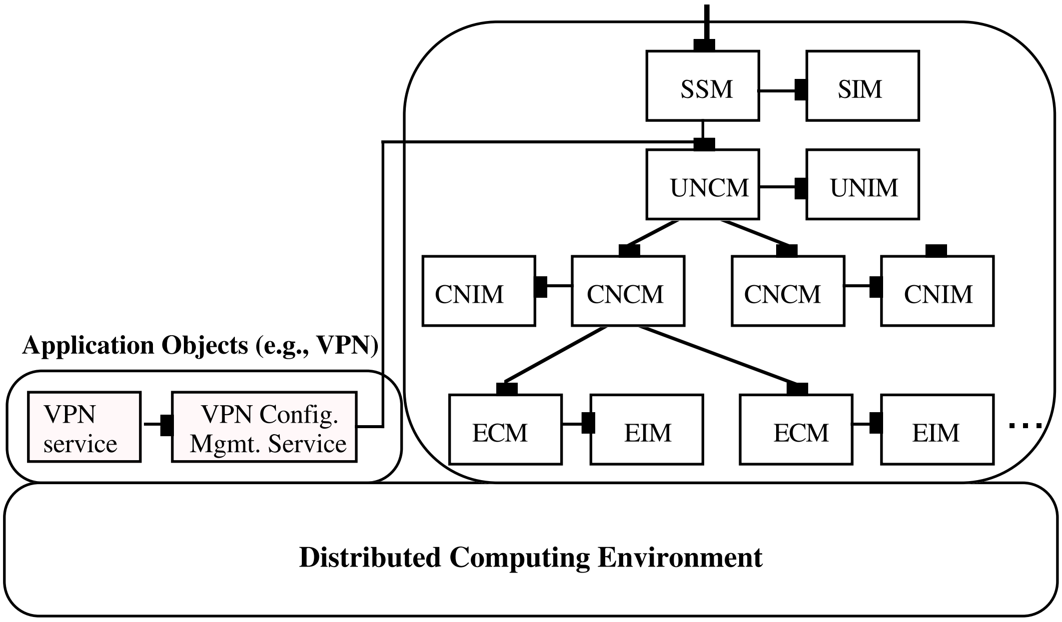


Figure 12: VPN Configuration Management with OAMS

Conclusion

In this paper, we have presented the service and network management architecture of the OAMS open service architecture. The basic components of this architecture are management information models encapsulated within computational objects. To specify and design these models, we have defined generic information structures consisting of three main entities (node, link, access point) and then instantiated them for the management of an ATM network consisting of ATM Fore Switches used to provide telecommunication services. The information structures proposed have been developed over DCE. They correspond to servers that are used by telecommunications services (e.g., virtual private network service) and management applications (e.g., connection or fault management service).

Then, we have designed and implemented the connection management function within OAMS that provides to telecommunication services, the establishment, release and management of calls and connections. This connection management function consists of a set of computational objects that interact with those encapsulating the information models, namely the information managers.

In [11] we have proposed an information model for managing multimedia resources (e.g., video codec) and in [24], an information model for managing a host (e.g., UNIX terminal). The next step will be to encapsulate these models into computational objects and integrate them into the OAMS telecommunications service architecture for managing the resources used to provide multimedia applications; moreover they will help allocating resources for etablishing, maintaining and releasing multimedia service sessions at the service level more accurately.

Acknowledgments

The author gratefully acknowledges the valuable help and comments of Falk Dietrich, Jean-Paul Gaspoz, Constant Gbaguidi, Frédéric Hess, Jean-Pierre Hubaux and Frédéric Torasso on this work.

References

[1] ITU-T Recommendations Q120x, Q121x, Q122x: The Intelligent Network, 1993.

[2] S. Znaty, C. Gbaguidi, J.P. Hubaux. "OAMS: An Open Management Architecture for Multimedia Services over ATM", Technical Report, EPFL/TCOM/SZ/95/01.

[3] M. de Prycker, Asynchronous Transfer Mode, Solution for Broadband ISDN, Ellis Horwood, Chichester, 1991.

[4] ITU-T. Rec. M.3010, "Principles for a Telecommunications Management Network", 1993.

[5] ISO/IEC JTC1/SC 21/WG7. "Draft Recommendation X901, Basic Reference Model of Open Distributed Processing - Part1: Overview and Guide to Use", June 1993.

[6] ITU-T Rec. M.3100,"Generic Network Information Model", Oct. 92.

[7] W.J. Barr, T. Boyd, Y. Inoue. "The TINA Initiative". IEEE Communications Magazine, March 1993.

[8] RACE project R1093, 10th Deliverable. "The ROSA Architecture, Release Two". Dec 1992.

[9] RACE Project R2049, 4th Deliverable, "CASSIOPEIA Open Services Architectural Framework". March 1995.

[10] J-P. Gaspoz. "Methodology for the Development of Distributed Telecommunications Services". Journal of Software and Systems, June96.

[11] C. Gbaguidi et al. "Management of Multimedia Resources: from a Generic Information Model to its Application to an MPEG2 Video Codec. IFIP INDC, Trondheim, June 96.

[12] ATM Forum, "ATM User-Network Interface Specification Version 3.1", Prentice Hall, 95

[13] A. Lazar, S. K. Bhonsle, K-S. Lim. "A Binding Architecture for Multimedia Networks". Available at http://www.ctr.columbia.edu/comet/xbind/documentation.html

[14] ISO 10165-1/CCITT X720: Management Information Model, 93.

[15] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen. "Object Oriented Modeling and Design". Prentice Hall, 91.

[16] ISO 10165-4/CCITT X722: Guidelines for the Definition of Managed Objects, 93.

[17] ISO 10165-7/CCITT X725: General Relationship Model, Jan 94.

[18] CFS-B.120. "The Concept of IBC and its Relationship to ISDN", RACE CFS B120, Dec 92.

[19] CFS-B.210. "Definition of IBC and Relationship to ISDN", RACE CFS B.210, Dec. 92.

[20] CFS-C. 110. "Methods for the Specification of IBC Services", RACE CFS C.110, Dec. 92.

[21] J-P. Gaspoz. "Object Oriented Specification and Design of a Service Management System on ATM Based Virtual Private Networks". PhD Thesis, Swiss Federal Institute of Technology, 95.

[22] Open Software Foundation. OSF DCE Application Guide, Prentice-Hall, 92.

[23] D. J. Hughes. "Mini SQL: A Lightweight Database Engine". Technical Report, Dec. 94.

[24] C. Gbaguidi et al. "Host Management: From Concepts to Application to the World Wide Web". Submitted to IEEE GLOBECOM'96.