On the Design of an Integrated Management Platform for an ADSL/ATM based Access Network using CORBA

F. Vandermeulen, P. Demeester,
H. Opstaele
Dept. of Information Technology (INTEC)
University of Ghent
Sint Pietersnieuwstraat 41
9000 Ghent
Belgium
{filip.vandermeulen,piet.demeester,
hans.opstaele}@intec.rug.ac.be

P. De Ceuleners, J. M. Reynders
Network Services Dept.
BELGACOM
Bld. E. Jacqmain 177
1030 Brussels
Belgium
{piet.deceuleners,
jean-marc.reynders}@is.belgacom.be

Abstract

In this paper, we address the problem of the end-to-end management of a heterogeneous public network. In the case study, the access part of the network is based on the ADSL technology for the provisioning of high speed IP over ATM to the customer and consists of ADSL network elements and ATM access switches. The core of the network is ATM based and provides connectivity to retailers, which in this case are Internet Service Providers (ISPs). We give an insight on the technological approach which consists in building a distributed platform, based on CORBA, that integrates existing element and network management systems and provides an end-to-end configuration, performance and fault management solution. The architecture of a generic resource adapter for SNMP devices as well as an automated SNMP to IDL mapping method will be detailed.

Keywords

Distributed Management, End to End Control, CORBA, TINA, ADSL, Access Networks, SNMP/CORBA gateways

1 Introduction

High speed network access, whether it is for residential or commercial purpose is one of the main driving forces for manufacturing high performance networking products and designing new software architectures both in the service management arena and in the network management arena. One of these promising high-speed access technologies is xDSL, which is taking benefit from clever multiplexing techniques such as DMT to transform the poor bandwidth of traditional copper wire into a high-speed multimedia data pipe. One of the popular members of the xDSL family is ADSL (Asymmetric Digital Subscriber Line), which is now boosting the market of high speed residential Internet access. ADSL has become a mature protocol and the first commercial ADSL products are now available. ADSL does not only bring a new technology but it will also change the commercial landscape of stakeholders providing Internet access. With the traditional analog modem technology it was mainly the ISP who terminated the access line on one of it's routers or modem pools and the same is true for ISDN. To this respect the Public Network Operator (PNO) provides a dumb data pipe through the PSTN which was actually not designed for data transport. Now, the network access product portfolio of leading ADSL vendors seems to shift the concern of Internet access back to the domain of the PNO. By putting an ADSL modem pool (an ADSL rack for short) next to the PSTN switch at the central office (CO), the data of the ADSL line can be split off and fed to a high speed ATM network within the administrative boundaries of the PNO. This ATM backbone network will aggregate the end-user data traffic and transport it to a router that peers with one of the backbone switches. Actually the first router which gives access to the Internet will function as a Remote Access Node (RAN) which is still under control of the PNO. The RAN will support for example the user in it's choice of an ISP (which is not possible now).

The ADSL stimulated return of Internet access control to the PNO domain also implies the need for a reliable management platform that enables the network operator to integrate the ADSL technology with it's installed base of equipment (the ATM network in particular). The network management integration problem accentuates itself in different domains:

- Configuration management: there has to be a consistent view on the structure of the network which is composed now of ADSL racks and ATM switches. This view needs to support a per user granularity which could bring a scalability bottleneck.
- Performance and Fault management: the alarm management and performance control, have to become integrated.
- Accounting management: the traditional PSTN billing system has to be integrated with the billing of the ADSL service. An important aspect with respect to this is that the traditional user-ISP contract will get split up into 2 peer-to-peer contracts: a user-PNO contract and a PNO-ISP contract.

In this paper we will mainly focus the first point of the list above, work is now also going on for the second point. To solve the configuration management problem we had to cope with different technologies (ATM and ADSL), different types of equipment, and different existing management interfaces. We applied a pragmatic approach for designing a management framework that allows to provide an end-to-end view on the network by using emerging middleware technologies such as CORBA and powerful design patterns such as those proposed by TINA-C.

The paper is further structured as follows: In section 2 we give an overview of the current network architecture and it's initial management situation. We then describe the rationale of the requirements imposed upon a unified end-to-end management system. Section 3 outlines the overall architecture of a CORBA based management framework, which overcomes the shortcomings enumerated in section 2. The off-the-shelf generic software component, the so-called Resource Adapter, which embeds the legacy equipment management interface into the CORBA distributed processing environment, is discussed in section 4. We will split up the discussion of section 4 in two parts. Firstly, we describe the basics of a new mapping from ASN.1-SMIv2 to IDL. Secondly, we will provide some details of the automated code generation tool which implements this new mapping. In section 5, we will address the design of the network management layer, which is based on TINA resource management. Finally, section 6 gives an overview of deployment experiences, and concludes the paper.

2 Staged "*CORBAtion*" of the network management architecture

2.1 Starting from a heterogeneous management situation

The network, which is at discussion in this paper, is heterogeneous in two senses. First, there is a diversity in networking technologies, being ADSL and ATM, but secondly different equipment vendors have provided the network elements. In the operational network the equipment of three vendors has to be unified in a coherent and unified resource management architecture, while a lab version of the implementation even added one additional vendor to cope with.

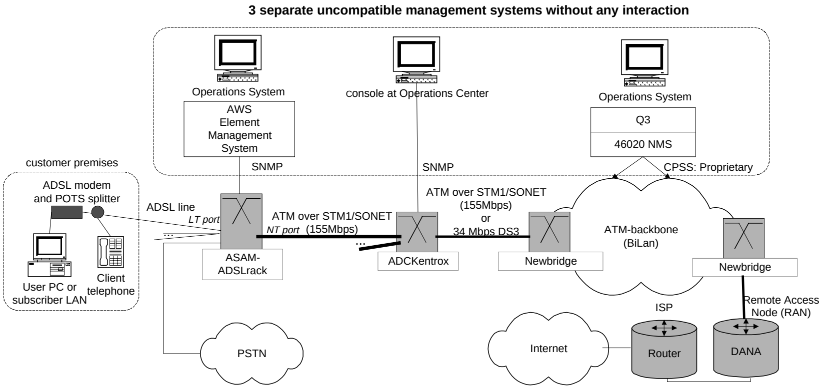

Figure 1 *Initial management situation: 3 highly incompatible EMS/NMS systems.*

Figure 1 gives an overview of the three vendor solutions, which have to be integrated in an enterprise wide management framework:

- The **access network** is ADSL based, and consists of ADSL modems at the subscribers end site and an ADSL Subscriber Access Multiplexer (ASAM) at the site of the Central Office. This equipment is based on the A1000 ADSL rack of Alcatel Telecom [14]. The fast or interleaved ADSL channels transport ATM cells from the modem to the ASAM where they enter an ADSL line termination (LT) and are multiplexed and framed into an STM-1 signal at an ATM over SDH output network termination (NT) port. The ASAM consists of 4 racks each containing 4 subracks filled up with 12 ADSL boards. One ADSL board provides 4 ports and terminates the digital subscriber line of 4 remote modems. The ASAM provides an SNMP management interface which implements the AToMIB [5] for the ATM management, and the ADSL standard MIB from the ADSL Forum for the subscriber line management. The vendor specific management solution for the ADSL access network (called the AWS) is mainly an element manager, which offers a graphical front end on the MIBs of the ADSL racks in the network. The application, whose capacity is limited to 200 ASAMs, communicates with the agent of the ADSL racks using SNMP.
- The SDH uplinks, streaming at 155 Mbps each, of different ASAMs are connected to one ATM **access concentrator switch** which gates to an ATM backbone network. The access switch of type ADC Kentrox AAC3 [15] is managed by SNMP. The existing management solution, which uses the SNMP agent on the firmware, is either a command line tool or a web based application using HTML and CGI.

- The **ATM backbone** consists of some high performance core switches from Newbridge. This vendor provided a Q3 interface (the 46020 solution [19]) which controls the network via a proprietary protocol (CPSS). On top of the Q3 exists a read-only SNMP interface, or a set of logging scripts that enable the check or inspection of configurations. The Q3 itself allows the connection management of the backbone.

The mission which has to be fulfilled by this high speed network, is the provisioning of end-to-end ATM paths on a per end-user basis. This path originates at the modem, runs through the ASAM as a VC cross connect, and enters a VP pipe that starts at an input port of the Kentrox. The VP tunnels through the Kentrox and the Newbridge backbone (called *BiLan*), and finally terminates on a router (called *DANA*), where the user VCs are terminated. The router provides access to one or more ISPs.
For the moment, the provisioning of the VP-paths over the access switches and the BiLan, and the configuration of the VC cross connects on the ASAM is fulfilled manually, by using the dedicated tools and management applications coming with each networking product. This means that the operations system staff has to control the resource management with three highly incompatible systems. These systems were simply not designed to cooperate for executing unified network wide transactions. The consequences of such a situation are:

- A *high cost of ownership*: Operations staff has to be educated for different management systems and technologies. This consumes a lot of time and costs a lot of money.
- Highly *incompatible information models*, with *lack of any central or decentralized network logic*. There is for example no central topology database and no centralized or coordinated alarm logging for error correlation. This makes off-line post-processing of performance information necessary.
- The concept of a *network wide transaction model* is completely absent. A transaction such as setting up an end-to-end path (VC or VP) happens by different people at different sites on different management tools. Communication takes place via email or phone calls to coordinate the connection parameters (e.g. VPI/VCI values).

From these disadvantages it is clear that this complex and dreadful management situation is not cost effective and error prone. Because of this, some studies have been undertaken to look how this situation could be overcome and which emerging technologies could bring an improvement in the field.

2.2 Requirements upon an integrated management design

Starting from the enumeration of the pending problems of the current situation we can sum up what is really required from a unified network wide management solution, independently of the used protocols or information models.

- The architecture should be able to cope with *different technologies* and when necessary *with different native management interfaces*. Most of the equipment provides an SNMP interface but some come up with Q3, CMIP-CMISE, TL1 or even a highly undocumented and obscure proprietary management interface.
- The framework has to be *highly scalable*, as we plan to provide thousands of subscriber lines in the access network. Real time responsiveness in the control of an access network with over 200 ASAMs (>100 000 subscriber lines) is crucial.

- The architecture is *highly transaction aware* and *network wide*. Two crucial transactions are envisioned. One is the setup of an end-to-end path for an end subscriber and another transaction is the consistency check of an existing line. Consistency checking involves controlling whether the connectivity parameters (VPI/VCI values and traffic descriptors) of a user path which are configured in the network are completely in line with what is stored in the subscription databases. In other words we want to have an end-to-end view on the connectivity topology of the network.
- The architecture copes with *dynamical configuration changes* such as insertion or removal of boards in or from a network element, addition or removal of a network element (e.g. an ASAM) and changes of the access topology (e.g. connection of a new ASAM to an existing Kentrox).
- *Pre configuration* of equipment (particularly the ASAM) is regarded as an important added value. With this, we target the feature of configuring an ASAM partly or completely without requiring that the device is already physically connected. This means that the planned configuration is stored in an off-the-shelf running database component and is afterwards downloaded when the device is physically in place.
- *Fault and performance management* are built in when the needs for these are there.

In order to bring all these advantages in one architectural management framework, we have taken advantage from distributed computing and off-the-shelf middleware software components. For us, CORBA was the enabling technology of choice to build a homogeneous software layer on top of a heterogeneous world of networking devices, where each of them comes with it’s own native management interfaces and APIs. The structure of our CORBA based solution is based on the design patterns of the TINA Network Resource Management Architecture and Information Models (TINA NRA/NRIM [3]).

The “*CORBAtion*” and “*TINArisation*” process has been executed in different steps each focusing at one particular requirement of the overall goal. This phased approach allows a smooth introduction of new principles and technologies in an already operational installed base of network equipment, and convinces operations staff in a gradual way. The first step was to provide an overall view on the static topology and connectivity (setup of paths) of the ATM/ADSL network. This only required *read* access on the equipment MIBs and because of this, deployment could succeed in a “ships in the night” approach. At this phase, the operators still configure the connectivity paths by using the individual management tools, but transaction based consistency checks produce a practical feedback on the network configuration stored in off-line databases. The second step was the introduction of end-to-end connection management over the complete network, requiring *write* access in the MIBs. At the time of writing, the third phase is still under development, and focuses at alarm and fault management (event notification management), auto-discovery of newly installed equipment, and pre configuration.

3 Functional and computational decomposition of the architecture

The overall architecture is depicted in Figure 2. At first, a middleware layer is introduced based on a CORBA software bus. The ORB daemons communicate over a Kernel Transport Network (KTN) provided by in band ATM PVCs running IP over AAL5. The cornerstone of the complete system is a generic gateway function that takes care of the control and management of a networking device. It acts as an *agent*

that knows all about the current configuration and conditions of the *adapted device*. This gateway is called a *Resource Adapter* (RA). The RA takes into account the setup and removal of connections on the subordinate network element. On startup, it learns the initial device configuration (present boards and ports, operational and administrative states) by uploading information from the MIB. The pool of RAs constitutes the *Element Management layer (EML)*. In next paragraphs, we will further elaborate on the specification and design aspects of a generic Resource Adapter.
On top of the EML a hierarchy of *Connection Performers* (CPs) has been designed in accordance to the TINA computational and informational decomposition. As the RAs provide similar abstract operations for ATM switches and ADSL racks, it is easy to build a CP-hierarchy on top of it. This approach gives the advantage of abstracting from the specifics of the technologies (ATM/ADSL), supports the end-to-end view of the network and enables the design of a scalable network management platform. The CP society has as fundamental task to support a scalable and automated transaction based connection management in the network. An end-to-end path (called a trail) is set up from an access port on an ASAM to a BiLan endpoint by requesting at the GUI system the setup of a trail on the "top CP". The CP-hierarchy constitutes the *Network Management Layer* (NML) which will be described in more detail in section 5.

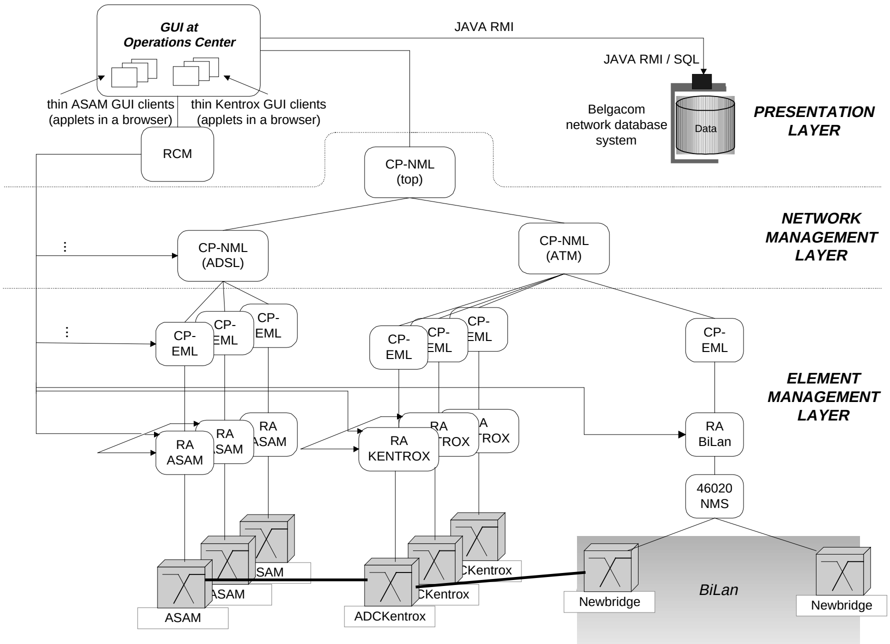

Figure 2 *Computational decomposition of the distributed CORBA based management architecture.*

The RCM (Resource Configuration Manager) takes care of several tasks. On startup of the system, the RCM boots and configures the RAs and the CP-architecture. During the lifetime of the system it acts as a daemon that listens on requests from the GUI OS to make or change configurations in the RAs and CPs. The RCM then routes the requests to the right component(s). The RCM does not need much persistency for doing this, it only needs a knowledge of the hierarchy of CPs and RAs.

The GUI system supports the front end of the system. It allows an operator to make and view configurations and facilitates decision processes.

The GUI is in close interaction with a database, which stores configuration data persistently. The database stores the following information items: configuration and identification (*LEXcode*) parameters of the network elements and the way they are connected to each other, location and type of boards configured in the network elements, description of the subscriber and the associated subscriber line. The database provides an ideal feedback instrument to support consistency checking on global configurations. On booting the management architecture, the RCM reads the database and extracts topology data and configuration data. Based on this information the RCM knows:

- For which network elements an RA has to be instantiated and how they have to be configured (e.g. board configuration).
- How the CP society has to be structured to handle end-to-end connection setups. Also the CP configuration (Topological Links and Link Termination Points, see section 5) can be easily extracted.

The database is accessed via Java RMI and an RMI/SQL gateway on top of the database. We will now go into a more detailed discussion for each of the layers composing the system.

4 The Element Management Layer

4.1 Architecture of a generic Resource Adapter

As remarked earlier, the EML is basically a pool of RA modules. Each network element is taken care of by a private RA, which abstracts the device in the Distributed Processing Environment (DPE). The RA provides a CORBA-IDL interface, which supports atomic and abstract operations that are generic and independent of the technology and the equipment vendor. The CORBA-IDL interface is subdivided in three parts, each taking care of a particular task: The first part is a *Configuration Management interface* (`i_ConfMgmt`) that allows the setup and deletion of PVC crossconnections. The second part is the *Device Configuration Interface* (`i_DevConf`), which allows to perform basic equipment configurations of the device. Actions of this category are for example: configuring profiles (e.g. *Access, CAC, traffic descriptors, ADSL*) and configuration or planning of boards and ports (e.g. planning an ADSL board which will be inserted later on in the field, and configuring the ADSL ports of each subscriber). Thirdly, there is the *Device Monitoring Interface* (`i_DevMon`), which allows to retrieve configurations and configuration changes from the RA and the device controlled by the RA.

Each of these interfaces (depicted in Figure 3) has interaction with an internal Knowledge Base (KB) in the RA that keeps track of the device configuration.

The KB has a generic core but can be expanded by inheriting internal classes and generic lists. The KB is introduced for two main reasons. First, it contains the right information to build up quickly a GUI snapshot of the device configuration at the presentation layer, so it improves responsiveness of the system because it acts as a proxy cache. Secondly, it allows to store pre configurations which can be downloaded later on into the MIB of the device. The KB, in conjunction with the upper interfaces and a pool of threads (see further on), is called the *Layer-2* of the adapter.

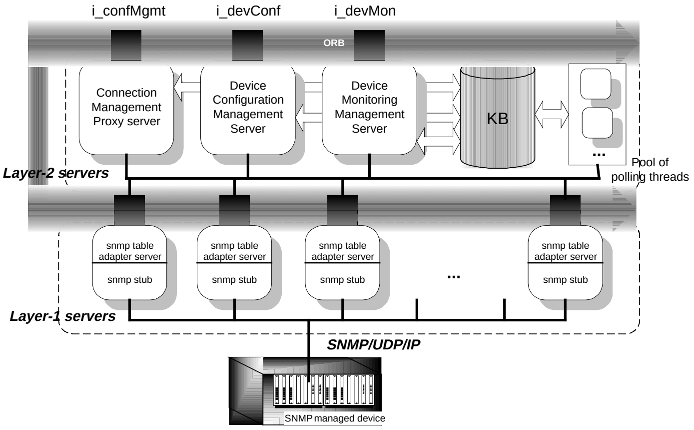

Figure 3 *Architecture, design and interface structure of the Generic Resource Adapter.*

The *Layer-1* of the adapter provides the basic communication protocol stack for the specific native management protocol. For the moment, most of the effort for this layer has been invested in SNMP because each of our devices offered an SNMP interface. Nevertheless, nothing inhibits us from introducing a Layer-1, which is based on CMIP-CMISE or TL1. The Layer-1 components are state-less protocol converters. They offer a CORBA-IDL interface on the MIB of the device, which supports GET, GETNEXT and SET operations on MIB variables simply by calling some IDL methods. The specifics of the management protocol and the PDU processing are hidden by these Layer-1 CORBA servers. To perform an abstract Layer-2 operation, a set of actions has to be executed on a group of Layer-1 servers. This is implemented as a transaction on Layer-2. If any of the performed sub-steps fails, automatic roll-back is executed. As such, the operations of each of the three interfaces at Layer-2 are implemented as transactions on the KB and the underlying interfaces of the RA.

Finally there is a pool of threads that calls directly on layer-1 methods in order to synchronize the KB with the device MIB. The thread pool performs regular polls (now every 30 seconds) to synchronize the information of the KB with the low-level information in the MIB. Polling is necessary because other managers could create some changes on the equipment configuration (e.g. the AWS or a craft terminal for the ASAM). The polling is ubiquitous to support the migration phase, as the multi-manager aspect is inherent to this phase. Remark that the threads only poll a small part of the MIBs. On the ASAM for example a set of readable bit vectors (`OCTET strings` syntactically) are at our disposal to get knowledge of the state of all boards, ports and modems. For example, there is a bit vector for administrative state, operational state, availability, and so on. By well tuning the choise of polled information, we can get an almost always up to date proxy KB, without spoiling a lot of KTN bandwidth.

The Layer-1 subsystem is generated by a tool (called the *Mib2RACompiler MRC*), which is especially designed for translating an SNMP MIB to IDL. The tool also aids in the design and code generation phase of the Layer-1 servers and the transaction processes at Layer-2. Next section explains the SNMP-SMIv2/IDL-CORBA mapping and the workflow of the mibcompiler.

4.2 Automated design of an SNMP/CORBA gateway: the MibCompiler approach

In this section we address the tool which has been designed to generate the Layer-1 and part of the Layer-2 of a generic RA. Currently the mibcompiler only supports the SNMP management protocol and it's associated MIB syntax ASN.1-SMIv2 [6,7]. Therefore the discussion is only for a SNMP based Layer-1.
On designing a SNMP/CORBA gateway, several approaches are possible. In [11] a comparison has been discussed between the *direct adapter approach* and the *adapter server approach* for designing a CMIP/CORBA gateway in the framework of TINA/TMN integration. For the SNMP/CORBA gateway, this twofold strategy exists in the same way.

- In a *direct adapter approach* an IDL interface is implemented on top of the internal device API. Still two possibilities for the implementation are open. Either you build an interface directly on top of the internal equipment software, or you provide an IDL interface on top of the SNMP agent. In the latter case, the SNMP services GET/SET/GETNEXT/TRAP, are provided in the IDL interface with variable length parameter lists (`IDL sequences`) for object identifiers and associated values. It is clear that the direct adapter approach requires access to the internals of the equipment software and this is not always possible. This approach also implies the provisioning of an ORB internally in the device.
- In the *adapter server approach* a separate on itself running server is implemented. This server receives CORBA invocations and translates them into PDUs, which are sent over a protocol stack to the agent in the remote device. Again two implementation possibilities are open. Either you make a static adaptation where you hard-code the MIB into the interface by providing an IDL mapping for each table or object, or you provide an IDL interface for the SNMP services as in the direct adapter approach. The adapter server approach does not require access to the code, but implies the provisioning of a protocol stack in the adapter, which introduces more latency and overhead.

In [11] the direct adapter approach has been followed where we developed an IDL interface with TINA semantics on top of a CMIP-CMISE–GDMO agent of a backbone ATM switch. For the mibcompiler however, we have used the *adapter server approach*. In comparison to the XoJIDM [9] mapping for SNMP/CORBA a somewhat different approach has been followed. From [11] we have learned that the XoJIDM mapping introduces a huge amount of objects running on the ORB. In some cases this is acceptable, but for an ADSL access network where over 1 000 000 objects have to be managed (almost 10 objects per subscriber line where “object” is an entry is some table), this would become a bottleneck. Because of this our CORBA/IDL translation has tried to reduce the amount of objects by specifying a mapping which reflects the scalar SNMP MIB structure more closely. The compiler maps every SMIv2 MIB table and every elementary object (which is not a table) on one IDL interface. As such an instantiation of the Layer-1 part of a RA for a device will run one server object for each table and each elementary object in the MIB. The functionality provided by the interface depends on the access policies of the columnar table objects or elementary objects (*elvar*). For read-only tables and read-only *elvars* a GET and a GETNEXT method are generated, for tables and *elvars* with read-write access a SET method is also generated. In the case of dynamic SNMP tables for which the row creation is ruled by a separate columnar object (e.g. the *RowStatus* column for SMIv2) an extra method is provided to perform SETs on this regulating columnar object.

```
atmVplTable OBJECT-TYPE
 SYNTAX  SEQUENCE OF AtmVplEntry
 ACCESS  not-accessible
 STATUS  mandatory
 DESCRIPTION "The Virtual Path Link
             (VPL) table."
 ::= { atmMIBObjects  6 }

atmVplEntry OBJECT-TYPE
 SYNTAX  AtmVplEntry
 ACCESS  not-accessible
 STATUS  mandatory
 DESCRIPTION
 INDEX   { ifIndex,  atmVplVpi }
 ::= { atmVplTable  1 }

AtmVplEntry ::= SEQUENCE {
  atmVplVpi            INTEGER,
  atmVplAdminStatus    INTEGER,
  atmVplOperStatus     INTEGER,
           …
  atmVplRowStatus      RowStatus
}

atmVplAdminStatus OBJECT-TYPE
 SYNTAX  INTEGER { up(1), down(2) }
 ACCESS  read-write
 STATUS  mandatory
 DESCRIPTION  "…"
 DEFVAL  { down }
 ::= { atmVplEntry  2 }


atmVplOperStatus OBJECT-TYPE
 SYNTAX  INTEGER {up(1), down(2), unknown(3)}
 ACCESS  read-only
 STATUS  mandatory
 DESCRIPTION  "…"
 ::= { atmVplEntry  3 }
```

```
interface i_atmVplTable {

   void add_entry(
      in  t_IndexOid indexOid,
      in  t_Integer atmVplRowStatus
    ) raises(snmpManagementError);

   void set_entry(
      in  t_IndexOid indexOid,
      in  t_Integer atmVplAdminStatus,
                       …
      out t_Integer atmVplRowStatus,
      in  t_apVector ap_policies
    ) raises (snmpManagementError);

   void get_entry(
      in   t_IndexOid indexOid,
      out  t_Integer atmVplAdminStatus,
      out  t_Integer atmVplOperStatus,
                       …
      out  t_Integer atmVplRowStatus,
      in  t_apVector ap_policies
    ) raises (snmpManagementError);

   t_EOTind getNext_entry(
      in   t_IndexOid inIndexOid,
      out  t_IndexOid outIndexOid,
      out  t_Integer atmVplAdminStatus,
      out  t_Integer atmVplOperStatus,
                       …
      out  t_Integer atmVplRowStatus,
      in  t_apVector ap_policies
    ) raises (snmpManagementError);

   void delete_entry(
      in  t_IndexOid indexOid,
      in   t_Integer atmVplRowStatus
    ) raises (snmpManagementError);
};
```

Figure 4 *A snapshot of the SNMP ASN.1-SMIv2 to IDL mapping of the* `atmVplTable` *of AToMIB (RFC1695).*

In Figure 4 a small snapshot is given for the mapping of the `atmVplTable` of the AToMIB [5] to IDL. This is a dynamic readwrite table ruled by the `rowstatus` columnar object. The figure depicts how the columnar objects are mapped on IDL types and how they are arranged in the to IDL converted SNMP operations. To access or set a columnar object of a table entry, the index in the table has to be provided. This is done in the `indexOid` parameter.

Figure 5 outlines the processing behavior of the mibcompiler. The compiler starts from a set of ASN.1-SMIv2 files ({`mib.txt`}) containing the description of the MIB and converts them to IDL (in a set of files {`mib.idl`}), mapping every table and every *elvar* on an interface. At the same time the compiler generates an implementation of the CORBA servers in a set of files {`mib.cc`} and {`mib.h`}. These implementation files contain the SNMP/CORBA stubs, which are responsible for constructing the SNMP PDUs and returning the results from SNMP protocol retrievals. Finally, the compiler generates the code for launching the servers (in a set of files {`mibsrvmainline.cc`}), and generates some scripts for registering the server implementations to the ORB. The set of MIBs that has to be compiled to support the functionality of the RA are grouped and browsed in the MIB organizer. The Layer-2 script editor allows to describe Layer-2 transactions in a generic way. These scripts allow a partial code generation of the Layer-2 servers. This code has to be refined by the implementor.

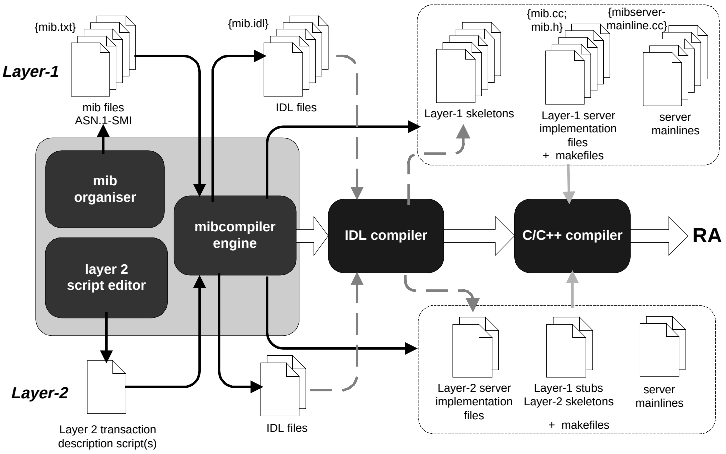

Figure 5 *Workflow diagram of the mibcompiler environment.*

5 The Network Management Layer: a lightweight TINA implementation

As explained in section 3, the CP architecture, which is designed as a distributed structure of servers, is needed to reflect the topology of the network and to support the setup of end-to-end connectivity (setup of VP and VC paths). This architecture, which is based on the TINA network resource architecture (NRA) [3], provides a scalable solution for PNO networks that typically represent one administrative domain. However, to suit the needs of our environment the NRA information model presented by [3] has been simplified and scaled down. The main reasons therefore were to avoid overkill and to enhance the performance of the system (reduce connection setup latency).

The information model contains two types of objects. Firstly, there is a set of objects that represent the topology of a layer network (physical layer, VP layer). This is called the static information of a layer network as it is not expected to change frequently. The second group of objects is used to represent connection setup in a layer network. This information is regarded as dynamic information for the layer network.

In the upper part of Figure 6, an OMT diagram is depicted of the static information of a layer network. The information elements depicted in the diagram are briefly reviewed below:

- The **Layer Network (LNW)** object and the **CPE** object: The Layer Network object represents in an abstract way the network layer we want to model. In our case this is the physical layer and the VP layer of the network.
- The **Subnetwork (SNW)** object: A Layer Network is composed of several subnetworks. A subnetwork is composed of one or more network elements which are of the same technology, the same type, or the same vendor or because they fulfill together one administrative policy. In our case different subnetworks have been setup for the two different technologies ATM and ADSL.

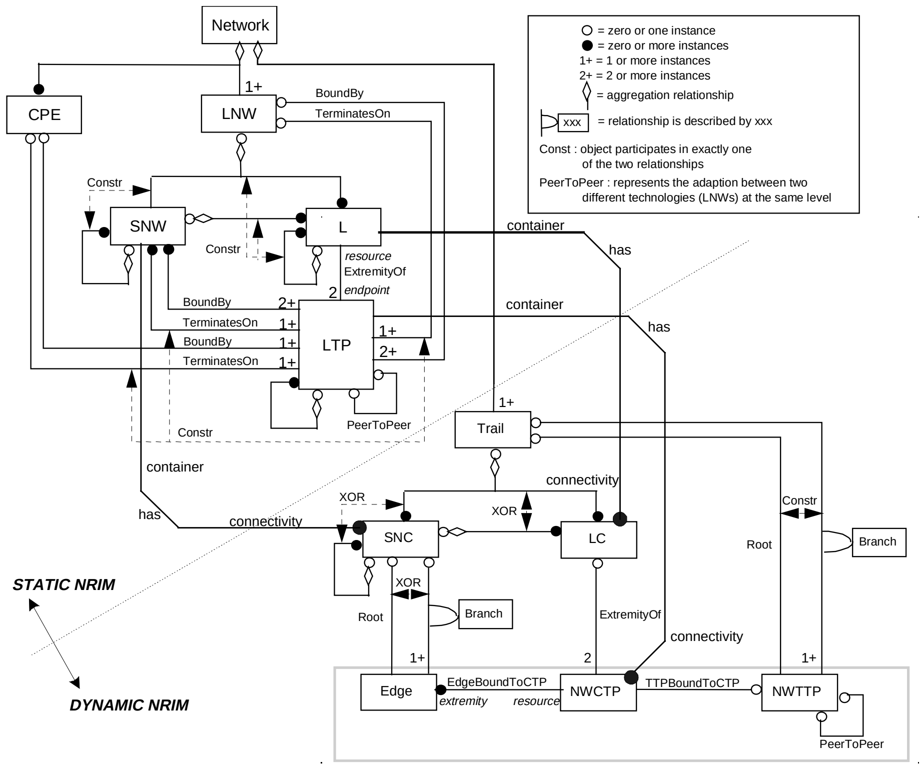

Figure 6 *OMT diagram of static and dynamic information model of network resource architecture implemented by the CP architecture.*

- The **Topological Link (L)** object: A topological link represents the binding that exists between two subnetworks. On the physical layer *L* corresponds to e.g. a SONET/STM1 link or a DS3 link between two VP or VP/VC switches, while on the VP-layer *L* corresponds to a VP path between two VC-switches.
- The **Link Termination Point (LTP)** object: An LTP represents an access point on a subnetwork. On the lowest order subnetwork (representing a single network element) an LTP is associated to a physical port (on the physical layer) or a VP termination (on the VP-layer).

A subnetwork contains a set of inner subnetworks in a recursive way, a set of topological links interconnecting the inner subordinate subnetworks and a set of LTPs at the edge. As such a subnetwork has a view on it's inner structure.

The lower part of Figure 6 depicts the OMT diagram of the dynamic information of a layer network. The information elements that were required for our needs are the following:

- The **Trail** object: A trail represents an end-to-end path setup over a layer network and is constituted as a concatenation of subnetwork connections and link connections.
- The **Subnetwork Connection (SNC)** object: A SNC is a connection spanning over a subnetwork from one access point to another. It is a concatenation of

subnetwork connections and link connections contained by subordinate subnetworks and topological links.

- The **Link Connection (LC)** object: A link connection is setup on a topological link between two subnetworks. It connects the subnetwork connections setup over the peering subnetworks.
- The **Network Connection Termination Point (NCTP)** object: Delimits an SNC
- The **Network Trail Termination Point (NTTP)** object: Delimits a trail and is bound to the NCTP of the originating and terminating SNC or LC of the trail.

Figure depicts an instantiation example of each of these objects for the case of our ADSL/ATM access network.

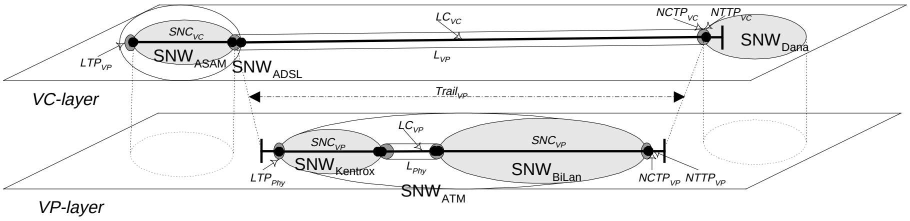

Figure 7 *Overview of the instantiation of TINA NRIM objects for an ADSL/ATM subscriber path.*

From the figure, it is clear that the CPs for the Kentrox switches and the BiLan will act as VP-CPs. They control the subnetworks of these devices and enable the setup of an end-to-end VP path over the ATM subnetwork (SNW_{ATM}). These VP trails become topological links (L_{VP}) in the VC-layer connecting the ADSL racks with the DANA router. The CPs for the ADSL racks act as VC-CPs and they control the subnetworks of the ADSL devices. A trail on the VC-layer (originating at the end-user modem, not shown in the figure, and terminating on the DANA router) represents an end-to-end subscriber path.

6 Deployment experiences and conclusion

Table 1 Overview of network elements, used platforms and deployment scale

Network Element	***Belgacom***	***ATLANTIS***
ADSL	Alcatel Telecom A1000	Alcatel Telecom A1000
ATM access switch	ADC Kentrox AAC3	IBM 8265
ATM backbone	Newbridge	IBM8285-NEC-FOREasx
Server – Tool	***Language***	***Vendor – Platform***
CORBA platform	IDL → C++ and Java	IONA Orbix & OrbixWeb
SNMP-API	C	CMU (Carnegie Mellon Univ.)
RAs / CPs	C/C++	Solaris
GUI / RCM	Java	Solaris / WinNT+browser
Deployment	***Belgacom***	***ATLANTIS***
# ASAMs	20 (3*12 ADSL-LT boards)	200 (12*12 ADSL-LT boards)
# Computing nodes	2 UltraSparc1 stations	4 UltraSparc5 stations

The software architecture has been proven by deployment at the site of Belgacom, the Belgian PNO, and has been tested extensively on the IP/ATM multimedia test bed ATLANTIS of the University of Ghent. Table 1 gives a summary of the hardware and software platforms used in both trials. At ATLANTIS we used both real ADSL network elements and a pool of software emulated SNMP test agents. The architecture proved that CORBA is an excellent technology to glue together incompatible management interfaces of heterogeneous networked systems. If realistic and efficient mappings from the native element management APIs (such as SMIv2) to IDL are used, together with powerful information models for the network management layer such as the TINA NRIM, then CORBA can scale very well for potentially very large networks.

Acknowledgment

Part of this work has been supported by the Flemish Government through IWT project ITA/950214/INTEC. We are very grateful to some people of TGX, the Belgacom Operations center, and to some people of Alcatel Telecom (Antwerp) for their advice on the A1000 system MIBs.

References

[1]. Yates M. et al, TINA Business model and reference points, version 4.0, May 1997
[2]. Abarca C. et al, TINA service architecture, version 5.0, June 1997
[3]. Steegmans F. et al, TINA network resource architecture version 3.0, February 1997
[4]. ITU-T Recommendation G.805, Generic functional architecture of transport networks, 1997
[5]. Ahmed M. and Tesink K., Definition of managed objects for ATM management using SMIv2, 1995
[6]. Feit S., SNMP, a guide to network management, McGraw-Hill Series on Computer Communications, 1995
[7]. Stallings W., SNMP, SNMPv2 and CMIP, The practical guide to network management standards, Addison-Wesley Publishing, 1995
[8]. Ghetie I. G., Networks and systems management, platforms analysis and evaluation, Kluwer Academic Publishers, 1997
[9]. Soukouti N., Hollberg U., Joint Inter Domain Management: CORBA, CMIP and SNMP, Integrated Network Management V, IFIP, Chapman & Hall, 1997
[10]. Vandermeulen F., Jean-Marc Reynders, Piet Demeester, TINTIN: TINA to Internet project, TINA-97, Santiago de Chile 1997
[11]. Reynders J. M. et al, TINA-TMN interworking: case study of an integration between the VITAL TINA platform and an existing TMN platform, IS&N 98, Antwerp 1998
[12]. Chan M. C., Lazar A. A., Stadler R., Customer management and control of broadband VPN services, Integrated Network Management V, IFIP, Chapman & Hall, 1997
[13]. Anerousis N., Lazar A. A., Scalable management services using Java and the World Wide Web, AT&T Labs Research, 1998
[14]. The A1000 ADSL rack, system description and MIB reference, Alcatel Telecom, 1997
[15]. The ADC AAC3 Kentrox MIB reference, ADC Kentrox, 1996
[16]. IBM ATM Workgroup Solutions: Implementing the 8285 ATM switch, IBM La Gaude, December 1996
[17]. IBM NWAYS 8265 documentation library, IBM La Gaude, 1997
[18]. ATM switch ATOMIS 5SE installation handbook, NEC, June 1997
[19]. Lamb R., CSM agents, release 2.0, release description, Newbridge Networks Corporation