Universität Karlsruhe
Fakultät für Informatik
Institut für Telematik
76128 Karlsruhe

A Flexible Management Gateway for ATM-based Networks

Hartmut Ritter
Frank Fock
Hajo R. Wiltfang

Universität Karlsruhe
Institut für Telematik

Interner Bericht 20/98
ISSN 1432–7864

September 1998

A Flexible Management Gateway for ATM-based Networks

Hartmut Ritter, Frank Fock, Hajo R. Wiltfang
Institute of Telematics, University of Karlsruhe
Zirkel 2, 76128 Karlsruhe, Germany
{ritter,fock,wiltfang}@telematik.informatik.uni-karlsruhe.de

Abstract

Network management is an important issue, especially for new and even complex network technologies such as ATM. At each time, state and operation of all components in the network should be controlled by the network management in order to have a global overview concentrated at one point and to react directly on detected problems. Unfortunately, the management interfaces of current ATM components are very different due to the ongoing standardization and vendor-specific issues. In this paper, we present the *ATM Management Gateway (AMG)* which can be classified as a proxy management agent. This gateway realizes a well-defined management interface for an ATM component, at which a minimal set of useful management information is provided. In order to adapt the gateway to the interface characteristics of given ATM components, it can be configured very flexible how to obtain the required information (e.g., which protocol has to be used and how information is identified). Moreover, this paper presents a *hierarchical architecture for the management of ATM networks* which consists of the three levels element management, management middleware, and management applications. The AMG entity represents a main component of the element management and the realized well-defined management interface builds the basis for the whole architecture. Finally, this paper describes how the implemented AMG entity has been successfully evaluated in an ATM testbed.

1 Introduction

The modern society is meanwhile characterized by a basic need for global communication. The focus here is not only the personal communication with help of telephones and the distribution of information based on radio and television. Additionally, the importance of computer-based communication increases more and more. However, computer networks of increasing importance and rising complexity require an adequate network management in order to monitor their correct operation and control their essential parameters. Moreover, the importance of network management increases additionally with the rising complexity of the underlying network technology, e. g., in the case of ATM. Furthermore, for new technologies such as ATM, much more development effort is put into the main functionality of the products and the management is often addressed with lower priority. This is additionally complicated by the not finished or still ongoing standardization in the ATM area. Based on the described facts, there is a great variety in the management functionality of today's ATM components. In detail, differences are given in the following areas:

- *Type of management interface* – The interface can be in-band (ATM) or out-of-band (an additional Ethernet or serial port).
- *Supported management protocols* – At the different interfaces, standardized management protocols such as SNMP (Simple Network Management Protocol [CFSD90]) or CMIP (Common Management Information Protocol [ISO91]) may be provided or other kinds of standardized protocols such as Telnet (text-oriented communication) or even special management protocols such as GSMP (General Switch Management Protocol [NEH+96]).
- *Available management information* – Standardized units of management information (e.g., ATM-MIB [AT94]) can be provided or, in most cases, only parts of these standardized units (e.g., only parts of the ATM-MIB). Moreover, a lot of vendors have defined special sets of management information for their own products.

Thus, applications for the management of one or even more components in a heterogeneous ATM network can only be developed with a great effort which results from the adapting overhead. Currently, available management applications for ATM networks or ATM components are designed for one dedicated component of one vendor or, in some cases, for a few similar components of the same vendor. However, much more useful are management applications that are suited for a lot of different components in a heterogeneous environment or for a whole ATM network.

In this paper, we present an architecture for the hierarchical management of ATM networks. Section 2 describes the three levels of this architecture. The hierarchical management architecture is based on a well-defined management interface which provides a minimal set of required information for each ATM component. In order to achieve this well-defined interface, the ATM Management Gateway (AMG) is proposed in section 3. This powerful and very flexible gateway acts as a proxy management agent for each corresponding ATM component. In section 4, we evaluate the ATM Management Gateway in a practical ATM testbed environment. To that aim, we used the ATM Network Monitor [Wil97, WF98] and the Web-based ATM Network Manager [Chr97] in order to compute and display the topology of the given ATM network and request additional information from components of the displayed topology. Finally, section 5 summarizes this paper and gives some ideas for future work.

2 Management Architecture for ATM Networks

In [Wil97], we proposed an architecture for the management of ATM networks in order to solve the problems caused by the heterogeneous situation of today's ATM components. This architecture consists of the following three levels (cf. figure 1):

- *Element Management* – At the base level, the element management deals with the management of individual ATM components. Thus, the element management is described by the specific characteristics of each particular component. These characteristics are related to the vendor of the component and the current development state as shown in section 1. An important objective for the element management is a well-defined interface to the management information that should provide a small set of absolutely required information. One solution to achieve such a well-defined interface is the use of proxy management agents as defined by the internet management model [CFSD90]. The proxy agent acts as a management gateway and translates the requests received at the realized well-defined interface according to the individual characteristics of the ATM component. At our institute, we have developed the ATM Management Gateway (AMG,

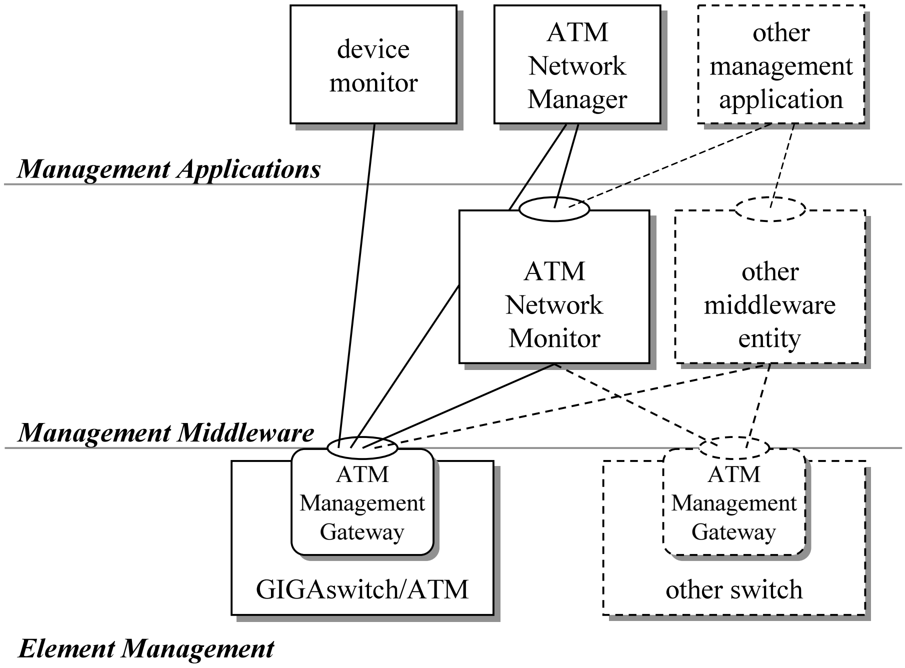

Figure 1: The Hierarchical Management Architecture

see section 3) for this task. The management interface provided by the ATM Management Gateway realizes a well-defined access to the management information for all entities located on the higher layers of the presented management architecture. At the management interface, we used the management protocol SNMP (Simple Network Management Protocol [CFSD90]) because of its great acceptance and wide deployment.

- *Management Middleware* – At the second level of the architecture, management middleware realizes services supporting the management of ATM networks. These services are more complex than the simple management functionality of the element management, since more or even all components of an ATM network are involved in middleware services. Usually, functional areas such as configuration, fault, and performance management can be addressed by the services of management middleware. For example, the ATM Network Monitor (ANEMON, [Wil97, WF98]) realizes a topology discovery for ATM networks and, based on the topology information, a monitoring of Quality of Service (QoS) for selected ATM connections. The main advantage of the middleware results from the concentration of important supporting services in one or some central entities which can be used by a lot of management applications. Thus, there is no need for these applications to realize the middleware functionality by themselves. In order to provide the supporting services management middleware uses the well-defined management interface of the underlying element management, which is realized with help of the AMG entity.

- *Management Applications* – At the third level of the architecture, management applications are located. They represent different kinds of front-ends to the management of ATM networks. Thus, management applications are generally characterized by a graphical window-based interface or a simple textual interface, at which the interaction with a human user is performed. The simple services of the element management as well as the more complex services of the management middleware are used by management applications in order to fulfill the tasks and actions requested by the human users. Because of the well-defined and component independent management interface of the element management, applications can be developed and used for different kinds of ATM components. In detail, management applications for only one single ATM component (not one specific) can be distinguished from applications for the management of more or even all components of an ATM network. [Wil96] gives an example of a device monitor for one ATM component (switch) and [Chr97] presents the Web-based ATM Network Manager which is designed to perform management tasks for a whole ATM network.

3 ATM Management Gateway (AMG)

In the presented approach, management gateways are used in order to provide the required information for management and monitoring of components in ATM networks (cf. figure 1). These components may be ATM switches as well as end systems. Each management gateway realizes a well-defined interface for a single network element. The entities of the management middleware and the management applications send requests to and receive responses from this gateway. Therefore, it hides the differences of the heterogeneous management interfaces of the network components from the entities of the higher layers of the architecture. It is the task of the management gateway to translate the received requests according to the characteristics of the individual network element and to forward them to the network component. Responses of the ATM component are re-translated and sent to the initiator of the request. Optimizations of this principle, caching of information e.g., will be discussed in section 3.2.

In order to realize this approach we propose the usage of proxy management agents acting as management gateways. The usage of proxy management agents has been defined by the internet management model [CFSD90]. As long as vendors of ATM components support different management interface types, proxy agents provide a simple solution. A proxy agent can avoid adaptations of the firmware running on the ATM components, otherwise the firmware has to realize the well-defined management interface. Moreover, a proxy agent can be realized on every open system. The higher layer entities of the management architecture only have to know the network and transport address of the agent. Therefore, a migration of the external proxy agent to a solution integrated into the ATM component requires no further change for entities of the higher layers of the management architecture. Only the address information has to be configured.

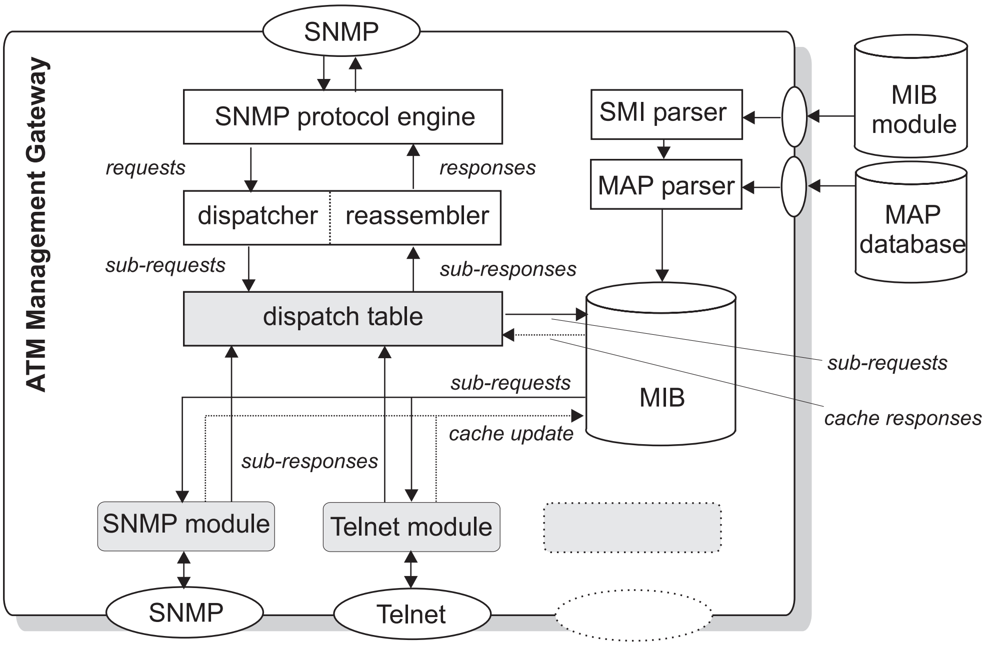

Figure 2: Structure of the ATM Management Gateway (AMG)

The structure of the ATM Management Gateway (AMG) is shown in figure 2. The structure will be described in section 3.1. Some basic design decisions should be discussed first.

Keeping in mind the heterogeneous world of ATM network components with many different interfaces and protocols (cf. section 1), an ATM management gateway hiding these differences should be highly configurable. This was realized by the modular structure of the AMG implementation. The communication modules realize the interaction between the management gateway and the different types of interfaces of an ATM component. These modules are each dedicated to one protocol. The support of different management interfaces of an ATM component can easily be done by integrating another communication module. Typical modules are a Telnet and an SNMP module. The modularity comes along with a wide configurability of the agent: Dynamic parsing of the MIB module and the MAP database at initialization time makes support of nearly every MIB possible. The incremental loading of MIB objects at runtime is nevertheless not implemented because it requires complex mechanisms and there seems to be little need for it by now.

At the interface to the higher layer entities of the management architecture, SNMP was chosen as management protocol for the following reasons: Both ATM Forum [AF96] and IETF [AT94] enforce SNMP as management protocol in ATM networks. The result is that most of the currently realized management interfaces are based on the SNMP protocol. Moreover, a lot of currently available management applications are also using SNMP. The information provided at the SNMP interface is specified in a MIB which is described in section 3.3.

The following sections 3.1 to 3.3 provide a detailed view on the structure of the AMG gateway, discuss caching strategies and present the proposed AMG-MIB.

3.1 Structure of the Implementation

In a first step, the proxy agent realizing the AMG must be initialized. Initialization of the proxy agent must be done because it can be configured to support different MIBs. The ASN.1-coded MIB module is parsed and spans up an internal object-oriented tree. The leaf objects of this tree represent information which can be retrieved from the ATM component by one of the specialized communication modules. Additionally, a relation between a MIB object and a method of retrieving its informational content is needed. Information about the retrieving method is - beside other information concerning e.g. the caching - given in the MAP database.

Now, the handling of an incoming request can be described: A request from a calling entity located in one of the upper management levels (cf.

figure 1) is received from the SNMP protocol machine and passed to the dispatcher module. The dispatcher breaks down eventually composed requests into single sub-requests, holds the request context in a list object, and issues sub-requests on the agent's MIB tree. The MIB tree acts as dispatch table: Each sub-request is mapped to a leaf object and according to the related information about the retrieving method, the sub-request is finally passed to one of the communication modules issuing a request to the ATM component. The response of the ATM component is passed directly to the reassembler, mapped to its context, and sent back to the calling instance using again the SNMP protocol machine.

Performance of the implementation could be improved by two ways: One item is the usage of caching discussed in section 3.2. Another item for a better performance is the generation of parallel processes with the unix system call *fork*. By this means, it can be prevented that the agent is blocked by requests of slow modules, e.g. a Telnet module. The usage of parallel processes allows for simultaneous handling of more than one request. Each sub-request issued by the dispatcher is realized as separate child process (shaded elements of figure 2). The responses are passed to the reassembler using message queues. The usage of threads is possible, but would endanger the portability of the implementation.

If there aren't any pending requests the proxy agent falls into the state of passive, interrupt driven listening at the UDP port of the SNMP agent. Only an update of the cached information as described in the next section is done after a given time interval.

3.2 Caching Issues

The proxy agent realizing the ATM Management Gateway and the built-in agent of the network entity will typically reside on different network nodes. Therefore, there must be a network connection in order to realize the communication between proxy agent and built-in agent. In this situation, two approaches to handle the communication are possible: Request-response and cache-ahead.

Using the cache-ahead paradigm, the proxy agent is periodically provided with current information by the related built-in agent of the network component. The proxy agent can therefore retrieve requested information elements without forwarding a request to the built-in agent of the network component. Only set requests must be forwarded directly. Using the request-response paradigm, all requests are forwarded to the built-in agent.

The proxy agent waits for the response of the built-in agent and finally forwards it to the requesting entity.

The advantage of the cache-ahead paradigm over the request-response paradigm is in general the reduction of requests forwarded over the network to the network component and therefore, a great reduction of response latency of the proxy agent. In our environment, caching of information is especially useful if information must be retrieved using slow modules like the Telnet module. Additionally, it is more efficient to read a whole table during such a slow Telnet session and to cache the values than to initiate a new session for every variable. The problem of the cache-ahead paradigm is that information might be out-of-date. Therefore, we implemented a hybrid approach: Some very time-sensitive values will not be cached at all whereas others will only be cached for a given duration. The caching options are part of the MAP database.

3.3 AMG-MIB

The amount of management information provided by the AMG entity for an ATM component is specified in one management information base (MIB). This AMG-MIB is the information base for all higher layer entities of the management architecture. Therefore, it should be kept universal and simple. An appropriate construction scheme is to specify groups of managed objects according to their level of indexing. Three groups can be identified this way [Rit97]:

- *System group* – This group contains information about type and configuration of the ATM network component as well as information about network and transport addresses. It provides an overall system view.
- *Port group* – The elements of this group are all indicated with one index, the port index. The group provides information about type and configuration of the ports of an ATM network component as well as information about signalling. Also included is information about neigbour addresses at each port. In conjunction with address information of the system group this information can be used to realize the topology discovery of the ATM Network Monitor middleware entity (cf. figure 1) [WF98].
- *Connection group* – Some entities of the management architecture, for example the Quality-of-Service module located on the ATM Network Monitor [WF98], need information about virtual connections in

order to track end-to-end connections. Therefore, the third group in the AMG-MIB provides information indexed in three levels: Port index, virtual path index and virtual connection index. Besides information about the local endpoints of the connections and address bindings, the connection group supports the setting and reading of traffic descriptors for each existing connection.

The prototypic AMG-MIB was fully specified and ASN.1-encoded and used in our evaluation environment described in the next section.

4 Evaluation of the ATM Management Gateway

An ATM testbed has been used in order to evaluate the operation of the implemented AMG entity. This testbed consists of a local ATM testbed at the Institute of Telematics and some components of the ATM network at the Computer Science department. Figure 3 shows the structure of the described ATM testbed. For each ATM switch in the testbed, an AMG entity has been instantiated and configured. In detail, two types of configuration profiles (MAP database, cf. section 3) have been created, one for the GIGAswitch/ATM from Digital and one for the VIVID CS 1000 from Newbridge. The AMG entity has only been used for the switches in the testbed, for the involved end systems (Alpha workstations from Digital), we used their built-in SNMP management agent.

As shown by figure 3, the ATM Network Monitor (ANEMON) middleware entity and the ATM Network Manager application have been used in order to evaluate the correct operation of the AMG entities in the testbed. The *ATM Network Monitor* is able to compute topology information for a given ATM network [Wil97, WF98]. To that aim, the ANEMON entity communicates with each ATM component in the network (i.e., the AMG entities for the switches, cf. figures 1 and 3) and figures out which switch or end system ports are connected with each other. All topology information collected by the ANEMON entity is store in a dedicated MIB, the ATM-TOPOLOGY-MIB [WF98]. Management applications can obtain this information by communicating with the ANEMON entity at its SNMP interface. The *ATM Network Manager* is a Web-based management application which uses the ANEMON information (cf. figure 1). On a Web-browser, the ATM Network Manager is able to display the topology information of the ANEMON graphically. This is achieved with help of a developed Java applet, the *atmTopolet*. In figure 4, the resulting situation

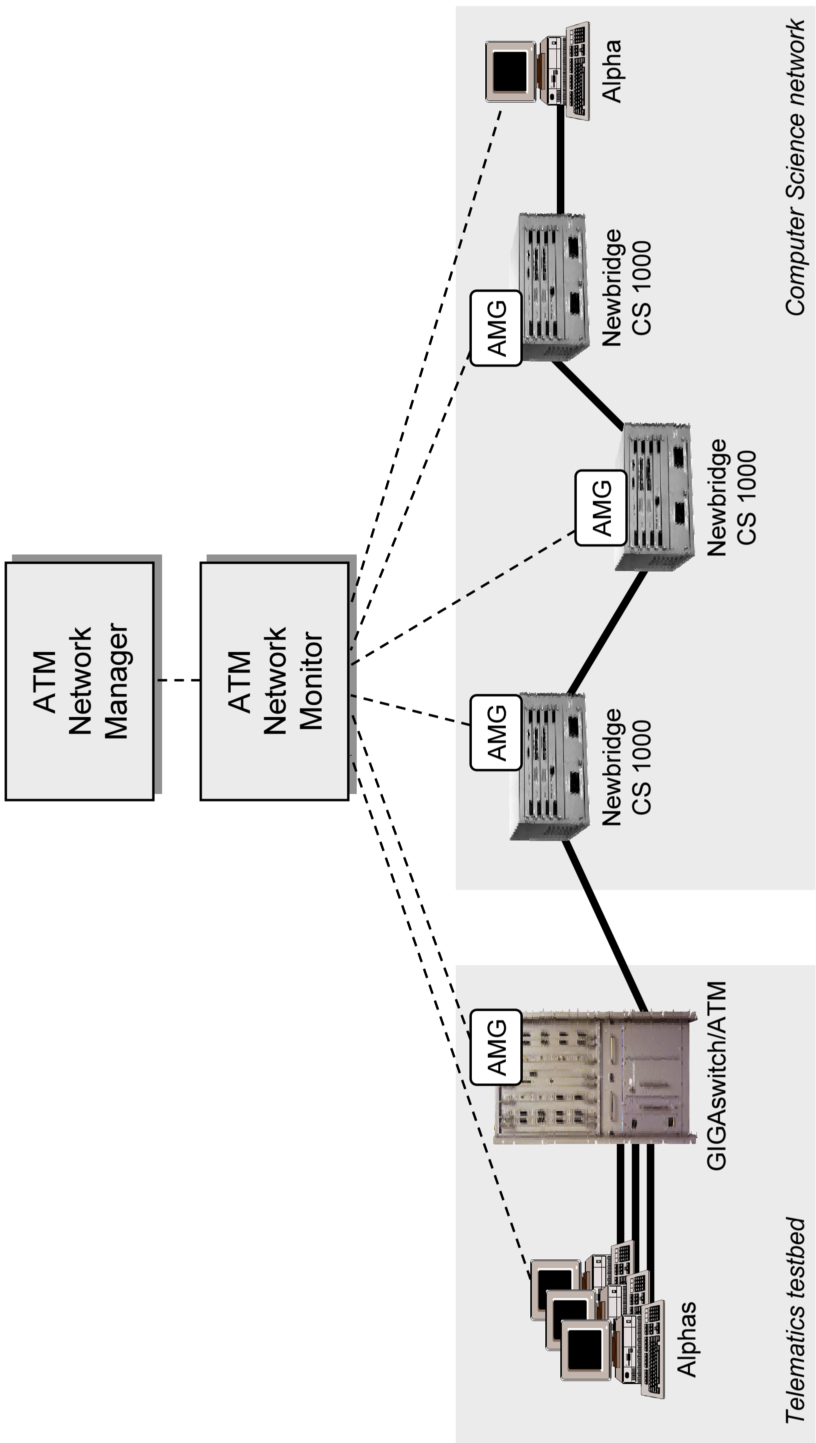

Figure 3: Testbed Environment for the Evaluation

for the testbed topology is shown. By selecting one component in the displayed topology, the user can get more and detailed information related to that component as show by the "Switch Menu" activated for one Newbridge switch in figure 4. For example, the cross-connect information is available for switches and interface information can be shown for end systems and switches. In both cases, the ATM Network Manager has to communicate directly with the ATM component (or its AMG entity).

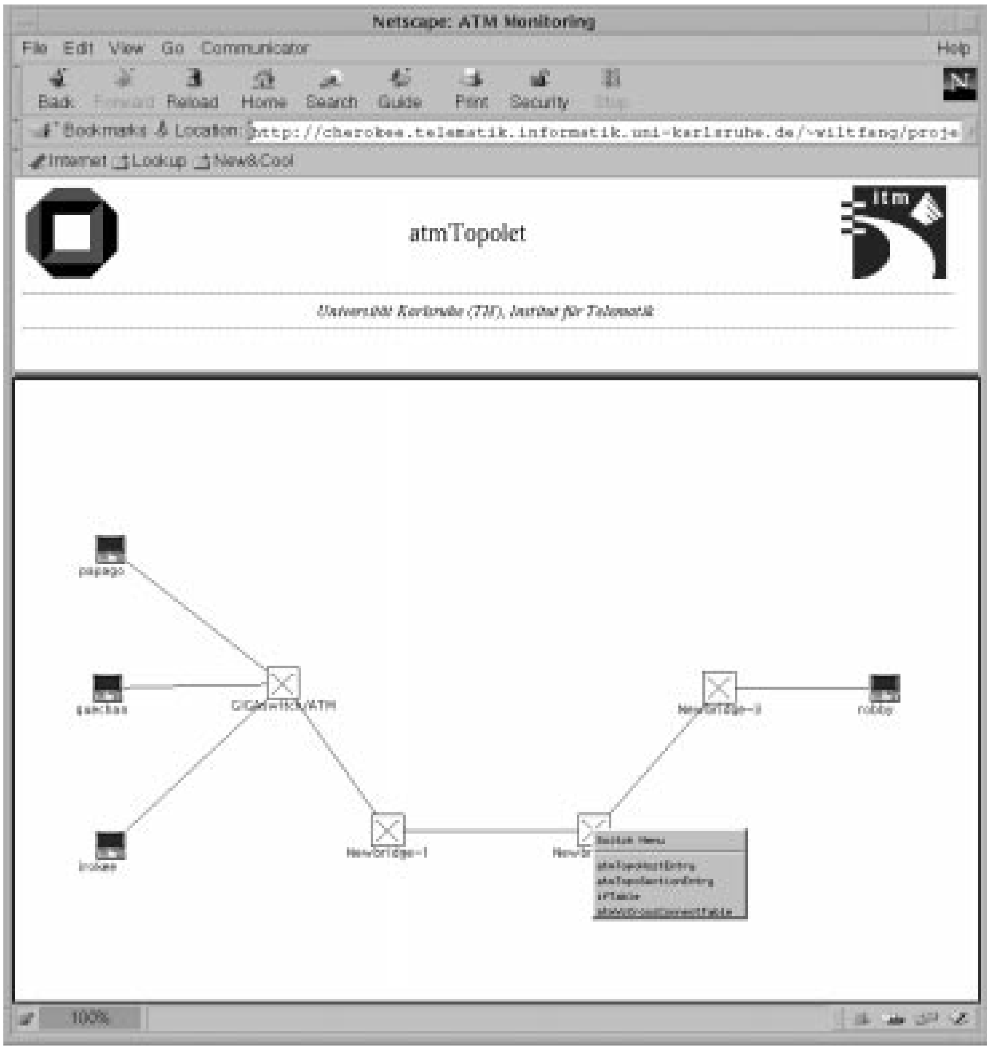

Figure 4: The Web-based ATM Network Manager

To summarize, the AMG entities had to operate together with the ANEMON middleware entity as well as with the ATM Network Manager in the performed tests. The ANEMON entity reads neighbourhood information from the port group of the AMG-MIB as well as general information from the system group. The ATM Network Manager communicates with a AMG entity when the user requests more information for one component in the displayed topology.

In our tests, the AMG implementation has shown flexibility and good performance. The flexible configuration possibilities made it very easy to adapt the AMG entity to the different switches. Both types of switches in the tests required the use of the SNMP and the Telnet module of the AMG entity in order to communicate with the corresponding interfaces of the switch. For the Telnet communication, the response time of the AMG could be drastically increased with help of the described caching strategy. However, especially for the Telnet-based communication between the AMG entity and the switch, some points should be mentioned because of the evaluation results. The design and quality of a Telnet interface may be very different. An important issue is the structure of the Telnet menu and its sub-menus as well as the navigating strategies within these menus. The Telnet client on the AMG entity has always to know the complete context of the current Telnet session in order to make the correct inputs. Moreover, for the same reasons, a deterministic and stable behavior is required for the Telnet interface of the switch. The SNMP-based communication between the AMG entity and the switch, however, did not cause any problems in our test and the delay introduced by the additional processing of the AMG was very small. No time-outs were detected for the response time at the SNMP interface of the AMG entity, when the requested information is available at the SNMP interface of the switch.

5 Conclusion and Future Work

In this paper, we presented a proxy management agent for ATM-based networks, the ATM Management Gateway (AMG). The AMG gateway realizes a well-defined management interface for any ATM network component, switches as well as end systems. It provides a minimal set of management information and hides the differences of the built-in management functionality of ATM network components. In addition, we presented a three-layered hierarchical architecture for the management of heterogeneous ATM networks. The AMG gateway represents a main component of the element management and thus, a basis for the architecture. The higher layer entities of this architecture rely on information provided by the AMG gateway.

Therefore, we enforced three items in designing the AMG: The described modular structure, coming along with a high configurability, enables the management of typically heterogeneous ATM networks. Performance issues were addressed in discussing and optimizing caching strategies. Finally, we presented the structure of a both simple and universal MIB of the AMG gateway. The overall performance could be proved in the presented testbed. The evaluations showed the advantage of the high flexibility of the proposed management gateway. The adaptation of the AMG entity to the switches of different vendors could be done very fast and easily.

Future work will be done to gain experience with managing ATM networks covering a wider area and consisting of much more intermediate and end systems. The integration of further communication modules into the AMG might be necessary as well as the adaptation of the proposed AMG-MIB to the needs of future integrated management tools. The development of management tools which provide the user with structured information about the network might also enforce some further optimization.

References

[AF96] The ATM-Forum. Integrated Local Management Interface (ILMI) Specification Version 4.0. af-ilmi-0065.000, September 1996.

[AT94] M. Ahmed and K. Tesink. Definition of Managed Objects for ATM Management Version 8.0 using SMIv2. RFC 1695, August 1994.

[CFSD90] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A Simple Network Management Protocol (SNMP). RFC 1157, May 1990.

[Chr97] J. Christ. Web-based Management of ATM Networks. Diploma Thesis (in German), Institute of Telematics, University of Karlsruhe, Germany, December 1997.

[ISO91] ISO/IEC. Information Technology – Open Systems Interconnection – Common Management Information Protocol Specification. International Standard IS 9596, 1991.

[NEH+96] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Ching Liaw, T. Lyon, and G. Minshall. Ipsilon's General Switch Management Protocol Specification Version 1.1. RFC 1987, August 1996.

[Rit97] H. Ritter. Design and Implementation of a Monitoring Interface for ATM Networks. Diploma Thesis (in German), Institute of Telematics, University of Karlsruhe, Germany, May 1997.

[WF98] H. R. Wiltfang and F. Fock. Topology Discovery for ATM Networks. In *Proceedings of the 9th IEEE Workshop on Local and Metropolitan Area Networks*, Banff, Canada, 17. - 20. May 1998.

[Wil96] H. R. Wiltfang. Monitoring the GIGAswitch/ATM. In *Proceedings of the 8th IEEE Workshop on Local and Metropolitan Area Networks*, Potsdam, Germany, 25. - 28. August 1996.

[Wil97] Hajo R. Wiltfang. An Approach to QoS Monitoring in ATM Networks. In D.W. Faulkner and A.L. Harmer, editors, *Proceedings of the European Conference on Networks and Optical Communication – Part II: Core and ATM Networks*, pages 328–335, Antwerp, Belgium, 17. - 20. June 1997.