Design and Implementation of a Web-based Tool for ATM Connection Management

Martin Bernhardt
mbhard@icsi.berkeley.edu

TR-96-041

August 1996

Abstract

At the *International Computer Science Institute* (ICSI), there is an ongoing effort to gain experience on ATM and multi-media applications. ICSI is participating in two ATM pilots called *Bay Area Gigabit Network* (BAGNet) and *Multimedia Applications on Intercontinental Highway* (MAY). Beside these wide-area trial ICSI's ATM network is used for local multi-media experiments. The ATM environment at ICSI is heterogeneous. Both, local and long distance traffic is based on permanent virtual connections. The management of this environment has often been cumbersome and time-consuming for a number of reasons: The ATM devices have to be accessed separately in an unintegrated manner. Different vendor-specific tools with different user interfaces are used. Configuration data is stored unstructured, redundant and not centralized. Users cannot setup or verify a connection without knowing device-specific details. Hence, the need for a software tool arose that can minimize the administrative work spent on connection management. This technical report contains my master's thesis which is about the design and implementation of *TOMCAD* - a **to**ol for **m**onitoring and **c**onfiguration of **A**TM **d**evices. Being a web-based software tool it can support local and wide-area connectivity and provide access for local and remote users.

Design and Implementation of a Web-Based Tool for ATM Connection Management

University of Stuttgart
Department of Computer Science

M A S T E R ' S – T H E S I S

by

Martin Bernhardt

International Computer Science Institute
1947 Center Street, Suite 600, Berkeley, California 94704-1198

August 1996

Contents

Chapter 1

Introduction and motivation

Today's computer networks inadequately address the needs of emerging multimedia applications with high-bandwidth and real-time traffic requirements. *Asynchronous transfer mode* (ATM) is often described as the technology that allows total flexibility and efficiency to be achieved in high-speed networks. Its promise of integrated services and wide-area connectivity has lead to a great deal of attention in recent years. Accepted in 1988 by the *International Communications Union-Telecommunications Standard Sector* (ITU-T) as the ultimate solution for *broadband integrated services digital network* (B-ISDN), ATM is expected to be the dominating technology for audio, video and data traffic in the wide area. Consequently, the communications community recognized ATM as a potential transmission technology not only for the public domain but also for private *local area networks* (LANs). The potential of coupling LANs via a public telecommunications network using the same transmission format forced the commitment to ATM among hardware vendors. In 1991, a group of manufacturers and users founded the ATM Forum in order to enforce the deployment of ATM in the private domain. A major task of the ATM Forum is the definition of standards for private ATM networks. The Forum tries to align these standards with those defined by the ITU-T for ATM equipment in telecommunications networks. However, detailed standards have not been released early enough to prevent hardware manufacturers from implementing proprietary solutions for the connection setup (signaling). Today, different vendor-specific signaling implementations exist. Signaling standards for the interface between private and public networks are still in process. Hence, currently ATM cannot provide universal connectivity without considerable management effort.

At the *International Computer Science Institute* (ICSI), there is an ongoing effort to gain experience on ATM and multi-media applications. ICSI is participating in two ATM pilots called *Bay Area Gigabit Network* (BAGNet) and *Multimedia Applications on Intercontinental Highway* (MAY). The BAGNet project is based on a high-speed *metropolitan area network* (MAN) in the San Francisco bay area. The funding by PacificBell ended in June 1996. In the second project (MAY) a transatlantic ATM link connects ICSI currently with a number of companies and institutions in Europe. The networks make host-to-host communication over long distances at multi-megabit transmission rates possible. However, prior to any data traffic between two ATM end-systems, a connection has to be established in between. Similar to the telephone network, there are two fundamentally different modes for connection setup.

1. on demand , by an internal signaling mechanism (dial-up line)
2. permanently, by an external mechanism (leased line)

Since the latter mode is static and inflexible, it would hardly be used to establish end-to-end communication between phones. Even for a small telephone network this would imply that a huge number of possible connections had to be maintained by means of network management. Assuming that every possible connection was setup permanently (fully-meshed network), the administrative overhead for fault handling and installation changes would still be high.

Unfortunately, wide area ATM pilots, like BAGNet and MAY, face exactly the scenario described above. Although ATM includes a signaling concept, it lacks signaling standards for the interface between private and public networks. Even for end-to-end connections within a local ATM network signaling can only be employed if all devices are inter-operable. This is usually not the case for heterogeneous environments that comprise ATM hardware from different vendors.

The ATM environment at ICSI is heterogeneous. Both, local and long distance traffic is based on permanent connections. The management of this environment has often been cumbersome and time-consuming for a number of reasons:

The ATM devices have to be accessed separately in an unintegrated manner. Different vendor-specific tools with different user interfaces are used.

Configuration data is stored unstructured, redundant and not centralized. Users cannot setup or verify a connection without knowing device-specific details. ICSI's multi-media projects will depend on a working ATM environment based on permanent connections for another several years. Hence, the need for a software tool arose that can minimize the administrative work spent on connection management. This master's thesis is about the design and implementation of *TOMCAD* - a **to**ol for **m**onitoring and **c**onfiguration of **A**TM **d**evices. Being a web-based software tool it can support local and wide-area connectivity and provide access for local and remote users.

The master's thesis is organized as follows:

Chapter 2 introduces the fundamental concepts of ATM networks and considers ATM connection management approaches with regard to their feasibility at ICSI. It also provides an overview of recent web-based network management concepts.

Chapter 3 describes specific features of ICSI's ATM environment and determines the functionality TOMCAD has to provide in order to support connection management appropriately.

In Chapter 4 the design of TOMCAD is described in detail. The system architecture is explained and all components and interfaces are introduced. The second and third section consider TOMCAD from a data- and function-oriented point of view.

Chapter 5 focuses on several aspects of the web-based implementation. The general procedure of function invocation is described and all elements of the user interface are explained in detail. The last two sections focus on the special solution for traffic monitoring and different methods of database access conclude this chapter.

Chapter 2

Background

Asynchronous transfer mode is a communications technology that utilizes a high-speed circuit-switching protocol. ATM offers the capability of high speed data transmission with minimum delay and guaranteed *quality of service* (QoS). This chapter comprises the technical background for the design of TOMCAD and the environment it was developed for. The first two sections summarize the basic terms, the connection paradigm and protocol issues of ATM. The third section introduces webs and web-based approaches for network management and contrasts these concepts to traditional network management concepts.

2.1 Asynchronous transfer mode

ATM transfers fixed size packets of 53 bytes (cells) in a connection-oriented mode. The cell-stream of a connection is asynchronous; different cell-streams are multiplexed on one physical link. ATM works by accomplishing two simple functions: hardware-based cell-switching and software-based connection-establishment. ATM offers the capability of data transmission with minimum delay and guaranteed *quality of service* (QoS).

2.1.1 Basic concepts

An ATM network consists of a set of interconnected ATM devices (Fig. 2.1)[1]. ATM hosts are workstations or PCs equipped with an ATM *network inter-*

[1]source: [1]

face connector (NIC). Unlike shared-media network technologies like Ethernet, ATM requires a switch to interconnect end-systems via point-to-point connections. A switch provides an array of ports to attach ATM end-systems or other switches. There are small-scale switches for private ATM networks (10-256 ports) and large-scale switches used by telephone companies for public ATM networks (up to 1000 ports). Different ports may support different media like unshielded twisted pair or coax copper cable, but usually fiber optical media is used since it allows higher transmission rates. ATM LANs can be connected to traditional LANs via ATM routers that translate protocols at the data link layer.

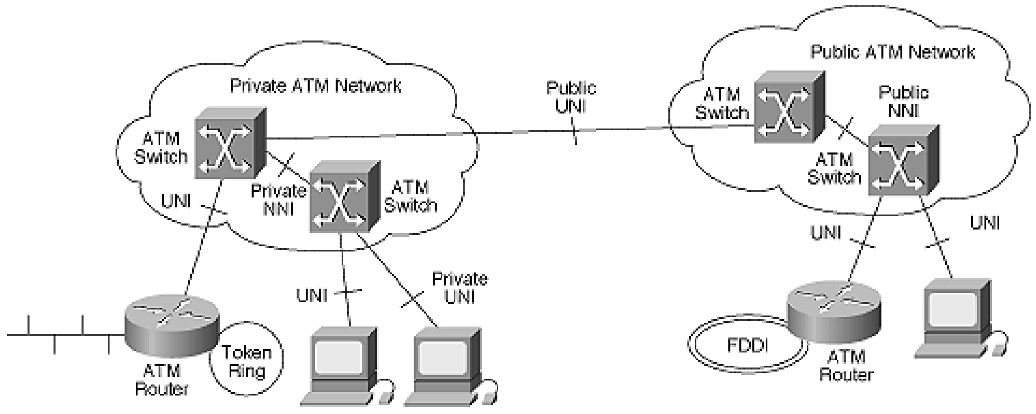


Figure 2.1: An ATM network

Two types of interfaces are defined for adjacent ATM devices:

- The *user network interface* (UNI) supports the connection between ATM end systems (hosts, routers) and switches.
- the *network node interface* (NNI) defines the connection between two switches.

Depending on the operator of the adjacent switch, public and private UNI and NNI can be distinguished. The institutions that have been working out and specifying these interfaces are the ATM Forum for private ATM networks and the ITU-T for the public domain. An ATM LAN has a meshed star-topology: the interconnected switches are meshed, building the core of the network. The end-systems at the UNI around this core form a star. This thesis focuses on ATM issues at the UNI.

ATM traffic is carried by fixed-size 53 byte cells. The 5 byte header of a cell contains fields for control information needed for cell-routing at the switches. The 48 byte remainder carries payload data. The VPI/VCI pair of the header (see Tab. 2.1) determines the route of the cell through the network. Predefined VPI/VCI pairs are used for cells that carry control information (e.g., for signaling or network management).

2.1.2 Virtual connections

ATM is fundamentally connection-oriented although fixed-length cells are switched across the ATM network. Cell-switching is implemented efficiently in hardware. Prior to the transmission of data a virtual connection has to be established between two end-systems. An ATM connection is called virtual since there is no dedicated line assigned to a connection. A virtual connection is compounded logically out of connection identifiers (carried by the cells) and routing tables maintained by the intermediate switches. A connection identifier is shared only among adjacent ATM devices along the path; it can change multiple times on the way to the destination system (Fig. 2.2). Table 2.1 summarizes the basic terms concerning virtual connections. Once a virtual connection is established, cells find their way as follows:

1. The sender generates cells with the VPI/VCI that is associated with a connection and transmits the cells to the adjacent switch.

2. The adjacent switch receives the cells, reads out the VPI/VCI field of each cell and looks up the corresponding outgoing port and the VPI/VCI pair in a connection table. A cell is discarded if no entry is found.

3. The switch generates cells with a new header containing the new VPI/VCI values and the 48 byte payloads of the received cells. It sends out the new cells at the outgoing port.

4. Steps 2 and 3 are repeated until the cells reach the destination end-system where they are passed to higher protocol layers.

For ATM signaling a one-path method of connection setup is used. An end-system that requests a connection generates a setup message that contains the destination address and the required traffic QoS parameters. The setup message is sent to the adjacent switch across the UNI. The switch

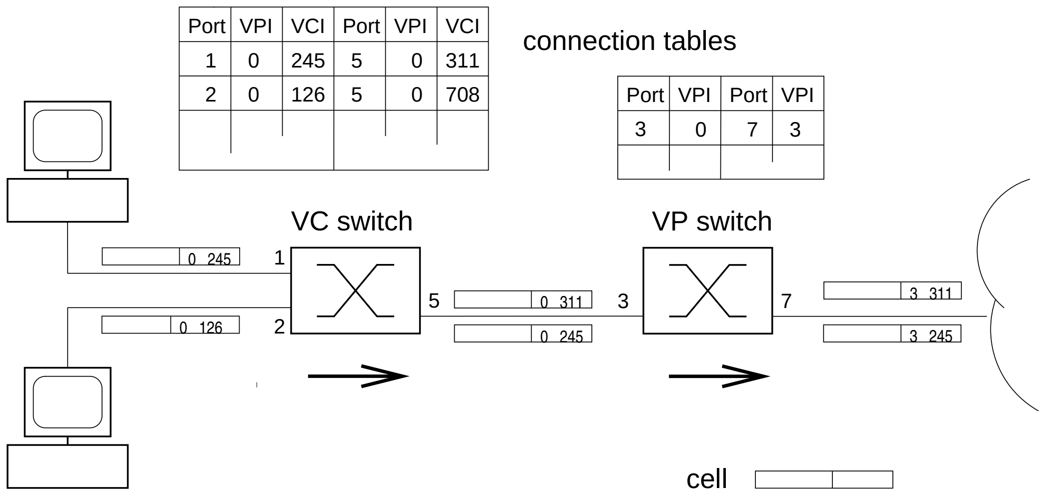


Figure 2.2: Cell switching

propagates the message through the network (NNI) using an ATM routing protocol, until the call reaches the switch adjacent to the destination system which transmits it across the UNI. The destination system either accepts or rejects the call. Cells carrying signaling requests have a predefined VPI/VCI pair (0/5). ATM signaling protocols differ by the type of interface. Signaling at the UNI is more simple since the request does not have to be routed there. Basic signaling at the UNI was standardized in 1993 by the ATM Forum as part of the UNI 3.0 specification. Standards for signaling at the private or public NNI have not been finalized yet. For this reason it is yet impossible to couple heterogeneous ATM networks in order to form a single large SVC-based ATM environment. Hence, at least for the near future, metropolitan or global area ATM networks will be based on PVCs.

Besides a routing protocol, signaling also requires an addressing scheme to identify the destination systems. The ATM Forum took into consideration the use of existing network layer protocols to establish virtual connections. These protocols have their own addressing and routing schemes that could be used for ATM addressing and signaling as well. This approach was called the peer model since the ATM layer is considered to be a peer of existing network layers. However, in order to decouple ATM from any existing protocol the ATM Forum defined an extra addressing and routing scheme. Network layer protocols have to be overlayed (overlay model), i.e. each ATM system is assigned an ATM address in addition to any network layer address. Hence, address resolution protocols have to be defined to map higher layer protocol addresses on ATM addresses. Three kinds of 20 byte address formats were defined for use in private ATM networks [1]. Section 3.1 outlines different ways to overlay the *internet protocol* (IP) on ATM.

2.2 ATM connection management

2.2.1 ATM layer management

The ITU-T defined the ATM reference model that consists of three planes (Fig. 2.3). These planes are divided further into several layers.

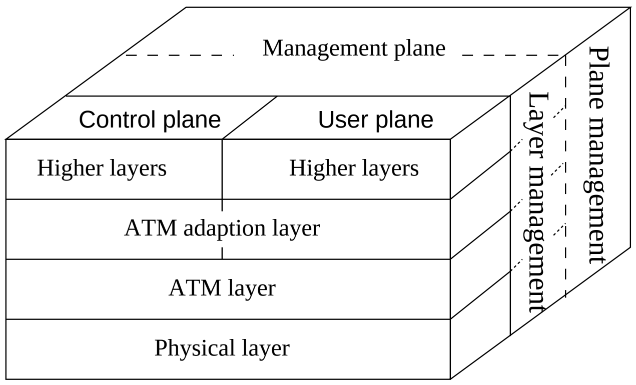


Figure 2.3: The ATM Reference Model

- the **user plane** consists of channels dedicated to transfer user data,
- the **control plane** consists of channels dedicated to transfer signaling information,
- the **management plane** consists of channels dedicated to transfer management information supporting the network management functionality, as well as the layer management operations.
- the **physical layer** defines a transport method between two ATM entities. It is responsible for the mapping of ATM cells to the underlying transmission system used for the correct transmission and reception of bits on the physical medium.

- The ATM layer is responsible for cell-relaying between ATM layer entities (hosts, switches, routers). This includes multiplexing and demultiplexing cells of different connections into a single cell-stream that is passed to the physical layer. It provides VCCs or VPCs with one out of a number of QoS classes. It interprets preassigned VCIs (like for signaling or network management).

- The purpose of the ATM adaption layer (AAL) is to provide a link between the services required by higher network layers and the services provided by the ATM layer. For B-ISDN the ITU-T defined four service classes according to different types of data traffic based on three connection parameters. In addition, different protocols were developed to support these service classes. Two ATM end-systems have to implement the same AAL in order to communicate. For end-systems in private LANs, AAL5 is recommended by the ATM Forum.

The functions at the ATM layer as specified by the ATM Forum for the UNI fall into two basic categories [2]. These are switching functions and management functions (according to the control plane and the management plane of the reference model). The management functions can be classified into fault management and interface local management functions.

- Fault Management
  Fault management functions are

  - Alarm surveillance
  - Connectivity verification
  - Invalid VPI/VCI detection

  These functions are accomplished using *operation, administration and maintenance* (OAM) cells to exchange related operation information. Five types of operational flows of OAM cells are defined at the UNI. Two of them perform functions at the ATM layer. Alarm surveillance consists of the detection, generation and propagation of VPC/VCC failures. Connectivity verification is an OAM function that allows a switch to perform a loop-back test on a link using a specific loop-back cell. An OAM cell can be specified to loop-back at any node throughout a connection. The third function is for recognition of cells with invalid VPI/VCI pairs.

However, - although useful for end-to-end diagnostics - operational flows concern virtual connections that are already established. Fault management can not be used for setup or setdown operations which are switching functions.

- Interface Management
  The ATM layer is required to provide some sort of local interface management information, i.e. configuration parameters of adjacent devices at the UNI. For the time until standards are finalized by the ITU-T for B-ISDN, the ATM Forum defined an *interim local management specification* (ILMI) [2]. ILMI supports bi-directional exchange of management information between *UNI management entities* (UMEs) (Fig. 2.4])[2]. The communication between two UMEs is protocol symmetric, i.e. each UME contains an agent and a manager application. The management information is stored in a *management information base* (MIB) at each UME. The ILMI MIB contains status and configuration information about VPCs, VCCs, address registration as well as ATM layer statistics. The MIB structure is separated from the access method which allows for the use of multiple management protocols. ILMI specifies the *simple network management protocol* (SNMP) on the AAL as the communication protocol using a pre-defined VPI/VCI pair (0/16). Moreover, the ILMI MIB can be accessed as well by general network management applications (e.g. from a *network management system* (NMS)) using SNMP over the *user datagram protocol/internet protocol* (UDP/IP) on the AAL. In this case, the peer entity is an SNMP agent, not the UME.

Unlike OAM flows, ILMI provides information concerning various parameters of virtual channels at the UNI. However, as with OAM flows, it was not designed to perform switching functions like connection setup. The ILMI MIB permits only read-access which prevents remote configuration of devices. However, being a generic information base for ATM hosts and switches, the ILMI MIB offers a vendor-independent way to access device parameters - at least for monitoring purposes. Sadly, many hardware vendors do not support it. The ILMI MIB will be reconsidered in Sec. 2.2.2.

[2]source: [14]

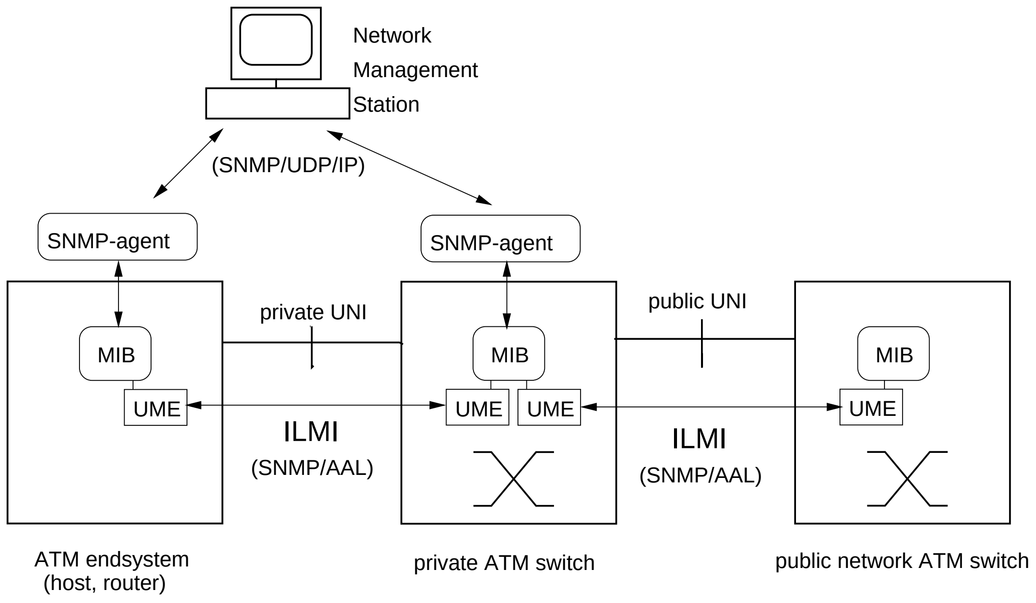


Figure 2.4: MIB access by UMEs and by an NMS

The ATM Forum defined a management framework that includes 5 key interfaces (M1-M5). They are essential to end-to-end monitoring and control. The framework takes into account the deployment of SNMP-based NMS. M1 and M2 define the interface between the management system at the customer site and the ATM end-station (Fig. 2.5)[3]. Standards for the M1- and M2-interface are not yet finalized.

Since 1988, there is an ongoing standardization effort of the ITU-T and other institutions around a communications concept called *telecommunications management network* (TMN) [7]. TMN defines a protocol-independent management framework for large interconnected public and private networks. It includes a five layer model for all kinds of management functions, ranging from vendor-specific hardware configuration to billing functions. TMN standards are designed to meet the needs of heterogeneous wide-area ATM networks. The TINA consortium, consisting of telecommunications operators, service providers, hardware and software vendors and research institutions world-wide, supports and coordinates collaborative research on TMN [17]. However, until standards are finalized and implemented, SNMP/IP based network management remains the first option for local, private ATM

[3]source: [9]

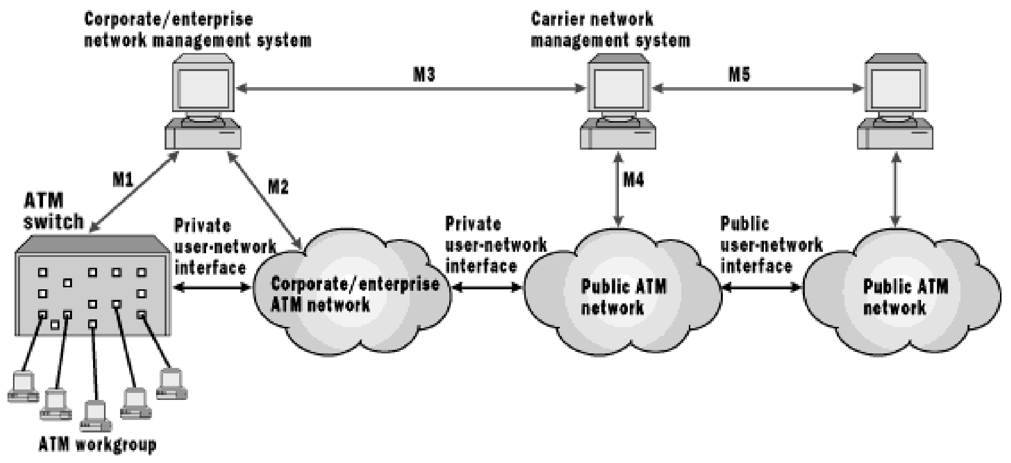

Corporate/enterprise network management system
Carrier network management system
M3
M5
M1
M2
M4
ATM switch
Private user-network interface
Corporate/enterprise ATM network
Private user-network interface
Public ATM network
Public user-network interface
Public ATM network
ATM workgroup

Figure 2.5: The ATM Forum management framework

networks.

2.2.2 The Simple Network Management Protocol for ATM

Standardized by the *internet engineering task force* (IETF) in 1990, SNMP is the dominating protocol for todays LAN management [12]. However, SNMP-based network management platforms like HP OpenView, SunNet Manager or IBM Netview basically cannot handle the end-to-end management of large wide-area ATM networks. Since these platforms are mostly based on a centralized network management architecture, their scope is limited to LANs. They do not support seamless end-to-end connectivity for thousands of users across high-speed LANs *and* WANs. In addition, SNMP was initially designed as part of the internet protocol suite for management of IP-based networks. SNMP cannot provide a view over a variety of conditions - from traffic throughput to cell delay to jitter - for a huge number of virtual connections.

Because SNMP separates the network management architecture from the architecture of the hardware devices, it provides an easy means to integrate different kinds of devices in an NMS. Management related information contained by elements in a network is defined in MIBs. An SNMP agent process -

residing at a network element - answers requests sent from an SNMP manager process (e.g. from the NMS). The manager process has no knowledge about how an agent or a MIB is implemented. It only needs to know the *structure of management information* (SMI) for each MIB. The SMI specifies the names, types and grouping of all accessible MIB objects. An MIB is specified by an extensible tree structure that defines names and types of managed objects. Subtrees refer to a group of managed objects with the same context. Vendors may choose to support generic MIBs - defined by standardization bodies - or to define their own MIBs that reflect special product-specific parameters. The MIBs themselves are numbered and grouped within a global tree structure organized by the *international standardization organization* (ISO). The Internet-subtree (internet(1)) - maintained by the - *Internet Assigned Numbers Authority* (IANA) - includes a node for private enterprises (Fig. 2.6)[4]. Any company can apply for a registered number in order to maintain its own MIB. Today, more than 1900 enterprises are registered.

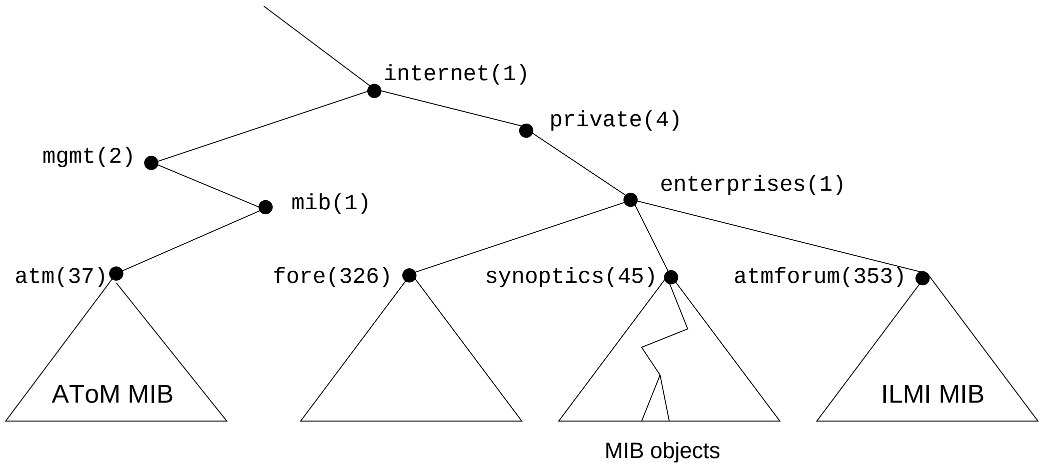


Figure 2.6: Part of the MIB hierarchy

The ILMI MIB covers a wide range of ATM parameters. However, many hardware vendors do not support it. Hence, depending on the vendor, names and types of MIB objects differ for the same type of parameter. In addition, vendors are not required to publish their MIB specifications. In this case, the

[4]The Fore-specific MIB object 'aal5ReceivedPDUs', for example, has the unique identifier 1.3.6.**1.4.1.326**.2.1.4.2.1.5

customer must use the proprietary SNMP software shipped with the hardware but can not develop his/her own management software to access the device. Some ATM vendors ship their products without any SNMP support and provide configuration and monitoring tools based on proprietary protocols and drivers. These tools only allow a device-based ad hoc management. They are not useful for an integrated connection-management.

In 1993, the IETF formed an AToM[5] MIB working group to develop the necessary management objects using the SNMP protocol for managing ATM devices. Among other MIBs, it based its efforts on the ILMI MIB. The current draft version of the AToM MIB defines object groups that can help net management applications construct certain views of the network. It specifies several SNMP management objects to manage ATM interfaces, ATM virtual channels, AAL5 entities and AAL5 connections. At the moment, the AToM MIB is used primarily to manage ATM PVCs. It does not define objects needed for ATM service management or for a true end-to-end view of the ATM network. One reason for this is that the MIB is limited to a single ATM end-system or switch and does not support the whole end-to-end span of a virtual connections. Thus the end-to-end connection management is achieved only by appropriate management of its individual segments in combination.

Although the AToM MIB does not feature the shortcomings of the ILMI MIB with regard to connection setup, its role will depend on whether the vendors are ready to support it. Facing the vast deployment of SNMP-based network management systems, AToM MIB appears as a promising step towards an integrated connection management - especially for PVC-based LANs.

2.3 Web- and Java-based network management

The tremendous growth of the Internet and the exploding number of IP-based customer premises networks (intranets) during the last years was driven by the enormous appeal of distributed hyper-media systems (webs). The

[5]The "o" in AToM is for *Synchronous Optical Network* (SONET) - a set standards for optical networks. ATM is often implemented on top of SONET.

simplicity of both, presenting and accessing information in a distributed, open and user-friendly way is the strength of webs.

However, the two fundamental concepts, the *hyper text markup language* (HTML) and the *hyper text transfer protocol* (HTTP), were not designed to implement generic client-server applications. The function of webs is information presentation rather than information processing. This will change, since HTTP and HTML are both subject to continuous improvements and web browsers and web servers are enhanced in terms of their ability to execute code. TOMCAD is a web-based tool that makes intensive use of a variety of recent enhancements. This section provides an overview of basic and advanced web-features and discusses their suitability for network management purposes. Moreover, products and projects will be introduced that are related to web-based network management.

2.3.1 Basic and enhanced web-concepts

Webs are hyper-media-systems providing easy access to distributed pieces of information including text, images, audio and video in an integrated, page-oriented manner. A client application (web browser) requests information from web servers using the HTTP protocol. Information is coded in HTML, a page-definition language which features simple, generic formatting statements (tags) [6]. The web browser parses the HTML-code in order to compile a web page according to the graphical capabilities of the client-system. A web page contains sensitive hyper-links that refer to other web pages or multimedia-objects at the same or another web server. Hence, web browsers provide comfortable and uniform point-and-click-access to remote systems. Unlike in traditional client-server systems, the client-program is decoupled from the contents of the application and can be employed generically.

The user interaction as described above is simple but restricted. Only static web pages can be downloaded but there is no means to submit user data in order to have the server process it. This changed with the HTML 2.0 standard released by the IETF. HTML 2.0 allows for programming web pages that include forms compound out of a variety of input elements like buttons, text areas, radio boxes, selection lists and pull-down menus. In addition, HTTP was enhanced appropriately to transfer the user-input. At the server-side, the *common gateway interface* (CGI) was defined). CGI defines an environment for executables CGI-binaries that reside at the web server. CGI-binaries process the user-data and - in response - generate a HTML-code that contains the results. HTML is subject to ongoing enhancements by both, the IETF and web software developers. The current standardized version is HTML3.0. Recent enhancements concerning layout capabilities are the notation of tables and of frames that can separate a browser window into distinct areas.

CGI provides a method to generate web pages dynamically. However, the processing workload is on the server-side. Since the client cannot control the execution of CGI-binaries, their scope is usually limited to short transactions like database queries. In 1995, Sun Microsystems introduced a programming language called Java [15, 16]. Since Java-bytecode is architecture neutral it can be interpreted on any machine that provides a Java-runtime-system (virtual machine). This allows for the implementation of mobile code as part of HTML-code (Java applets). Web browsers that feature a virtual machine can download and process Java Applets. Once downloaded, applets are executed and controlled at the client-side without further server-communication. Script-code as part of HTML-code is another extended HTML-feature. The code is interpreted by the browser and can control user-interaction and process user input to a certain extent.

2.3.2 SNMP and HTTP compared

Web servers on managed devices and web browsers as managers could play the role of SNMP-agents and SNMP-managers, respectively, in order to perform network management tasks. This section discusses benefits and shortcomings of HTML and SNMP.

- open protocols
  Both SNMP and HTTP are IP-based protocols that are standardized by the IETF and widely accepted and deployed. Hence, both support open concepts.

- asymteric protocols
  Both SNMP and HTTP are asymetric. While SNMP agents can initiate a communication (traps) - e.g., to send an alert - web servers can only respond to requests.

- handling output
  An SNMP manager collects raw data like counters and strings. It can

aggregate and convert data for different purposes (graphical presentation, storage in a database). Web browsers can only parse and display the HTML-code they receive; they are unable to process it in any other way.

- handling input
  In SNMP-based environments the manager application determines how a user can supply input. This could be a graphical user interface, a text terminal or a command-line interface. In the web-based case, it is the HTML-code sent by the web server that defines the layout of input forms.

- structure of management information
  HTTP does not include a MIB/SMI concept that would allow for developing its own management applications. Only web browsers can be used to access a managed device. Integration into existing management systems is not possible without interfacing software.

- views of the network
  Different SNMP-managers can aggregate retrieved data and present an integrated view of the network. A web browser, however, can only display one managed device at a time.

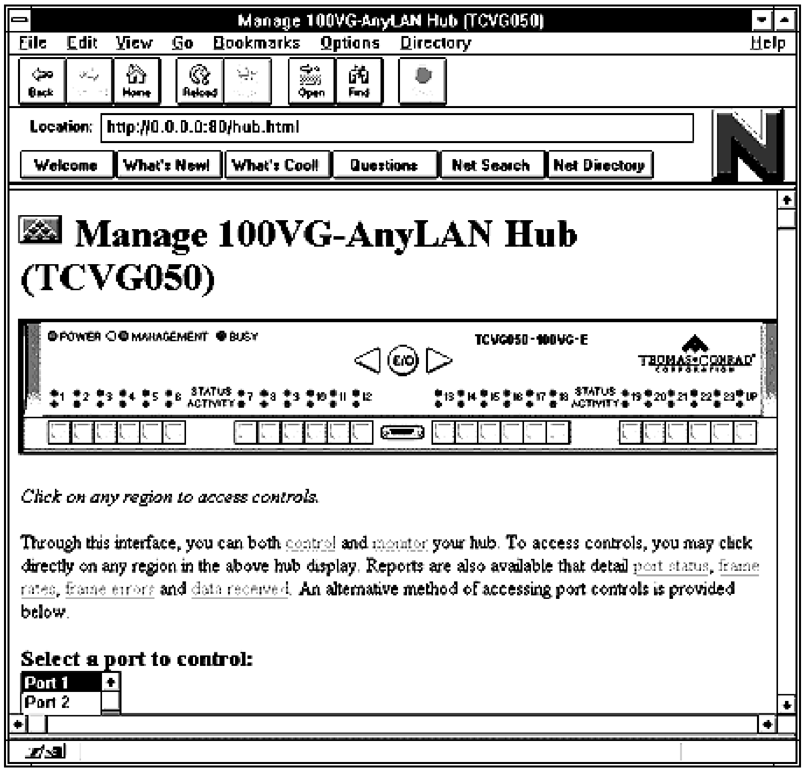
- remote access
  Web servers grant easy access from any host of an intranet, assuming a web browser is available. This is not the case for SNMP-managers which require that a client-front-end is installed.

- security
  Web servers can be configured for access over the Internet. This can be useful as well as a security risk. SNMP over the Internet is unusual but possible. Again, the manager station has to be equipped with appropriate software (see Sec. 5.4).

- costs
  Web servers instead of SNMP-agents can cut the costs for the development of configuration software since no client front-end has to be implemented. In addition, management staff can be decreased since remote management is possible.

2.3.3 Practical examples

As the comparison in the previous section shows, HTTP can not be considered a substitute for an integrated LAN management. However, the following examples prove that web-based management software can be useful for device configuration and as adjunct with SNMP-based software [9, 10].


Manage 100VG-AnyLAN Hub (TCVG050)
File Edit View Go Bookmarks Options Directory Help
Back Home Reload Open Find
Location: http://0.0.0.0:80/hub.html
Welcome What's New! What's Cool! Questions Net Search Net Directory
POWER MANAGEMENT BUSY
TCVG050 - 100VG-E
THOMAS-CONRAD
STATUS ACTIVITY

Manage 100VG-AnyLAN Hub (TCVG050)

Click on any region to access controls.

Through this interface, you can both control and monitor your hub. To access controls, you may click directly on any region in the above hub display. Reports are also available that detail port status, frame rates, frame errors and data received. An alternative method of accessing port controls is provided below.

Select a port to control:
Port 1
Port 2

Figure 2.7: The Webcheck user interface

- Webcheck by Tomas-Conrad, Corp.
  Thomas-Conrad, Corp. (Austin, Texas) is one of several vendors adding web-based management software to their networking products. For their hub TCVG050, they offer a configuration tool called Webcheck - a proprietary management agent embedded in the hub's management processor. The user logs in to Webcheck using a browser.

  Once connected, the browser retrieves a graphic depiction of the hub and its status (Fig. 2.7)[9]. By pointing and clicking on the picture of the hub, net managers can configure ports, look up the status of a given port, and receive usage statistics on a per-port basis. Another company, Tribe Computer Works Inc., developed similar software for its IP-routers.

- Cabeltron's approach
  Cabletron's SNMP console 'Spectrum 4.0' is equipped with a report generator that converts management data to graphic files in *graphics interchange format* (GIF) and stores them on a stand-alone web server (Fig. 2.8)[9]. Although network managers can read these images remotely, there is no way to control the network via the browser interface.

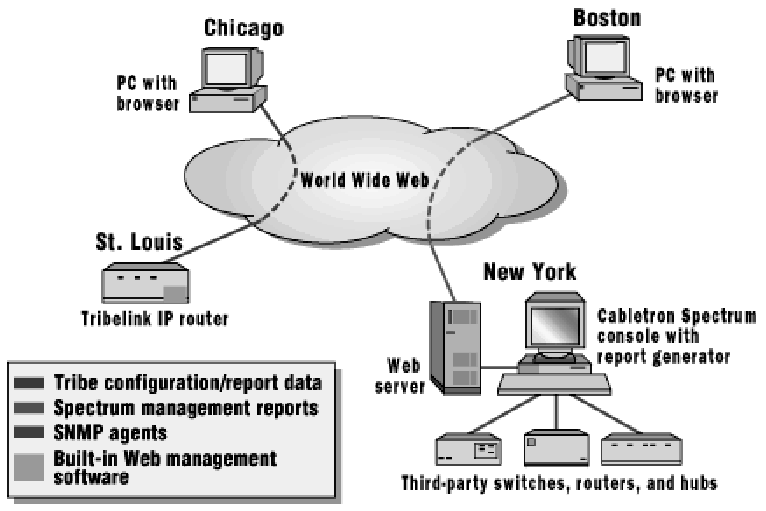

Chicago
PC with browser
Boston
PC with browser
World Wide Web
St. Louis
Tribelink IP router
New York
Cabletron Spectrum console with report generator
Web server
Tribe configuration/report data
Spectrum management reports
SNMP agents
Built-in Web management software
Third-party switches, routers, and hubs

Figure 2.8: Direct and indirect device access via the web

- Advent Network's Java SNMP package
  Advent Networks, Inc. provides another software concept to bring HTTP and SNMP together. This company released a Java SNMP package that can be used to develop Java Applets for network management. This allows implementing SNMP managers within a web page and hence, web-based access to SNMP agents. The package is used for traffic monitoring purposes within TOMCAD (see Sec. 5.4).

Term	**Description**
VC	A *virtual channel* describes a unidirectional flow of ATM cells between connected ATM devices (switches or end-systems). Both systems share a common identifier for a certain virtual channel.
VP	A *virtual path* is a bundle of VCs all of which are switched transparently through an ATM switch. The concept of VPs allows for mapping a set of VCs without changing the VCIs.
VCI	A *virtual channel identifier* is a 16-bit value in the cell header. Together with the VPI it identifies the virtual channel that carries the cell
VPI	A *virtual path identifier* is an 8-bit value in the cell. header. Together with the VCI it identifies the virtual channel that carries the cell.
VCC	A *virtual channel connection* is a concatenation of VCs between ATM end-systems.
VPC	A *virtual path connection* is an aggregation of VCCs.
PVC	A *permanent virtual connection* is a VPC or VCC set up by an external mechanism and not by signaling. Establishing a PVC includes setting up a concatenation of VCs between the end-systems values and allocating resources (bandwidth, QoS parameters) at the intermediate switches. The PVC is permanent since it is independent on whether the connection is used for data traffic or not. Hence, a PVC is comparable to a leased telephone line. To delete the connection the resources have to be released explicitly. PVCs always require some manual configuration. As such, their use can be very cumbersome. In order to avoid the signaling overhead, PVCs can be used to interconnect switches and/or routers in a static manner.
SVC	A *switched virtual connection* is a VPC or VCC that is set up automatically by a signaling protocol. The signaling protocol also deletes the connection when it isn't used anymore. Hence, SVCs are similar to dial-up telephone lines. SVCs are rather used for short-term connections between ATM hosts. Different signaling protocols have been defined by hardware vendors because of the lack of standards. Interoperable hardware within one ATM network can make the use of SVCs impossible.
connection table	Each switch maintains a *connection table* that maps an incoming VC to an outgoing VC. An incoming VC is uniquely identified by the VPI/VCI header of the cells it carries together with the number of the port that receives the cells. The connection table maps these three values to the port number and the VPI/VCI pair of the adjacent VC.

Table 2.1: Terms related to virtual connections

Chapter 3

Requirements for a connection management tool

The first section of this chapter introduces specific properties of ICSI's ATM environment that affect the design of the connection management tool. These are the heterogeneous ATM hardware and software, ICSI's local and wide-area ATM trials and the networks on which they are based. The properties give rise to several requirements for TOMCAD; they are outlined in the second section.

3.1 ATM infrastructure at ICSI

The ATM infrastructure at ICSI comprises many different hard- and software components. The connection management tool has to handle this heterogeneous equipment. The following list provides an overview.

- workstations
  ICSI uses currently 9 of its UNIX workstations as ATM hosts. These are SPARCstations of type 2, 5, 10 and 20 that run the SunOS4.1.3, Solaris2.4 and Solaris2.5 operating system. There are different versions of configuration tools for different operating systems versions.

- ATM network interface cards (NICs)
  The SPARCstations are equipped with an SBus ATM NIC by Fore or Interphase. These are the ForeRunner SBA-200E NIC and the 4615 NIC by Interphase.

- ATM switch
  ICSI owns one LattisCell-10114SM switch from SynOptics. It has 14 multi-mode-fiber ports (155 Mpbs) and 2 single-mode-fiber ports (45 Mbps) for external connections. A 10BASE-T Ethernet port connects the switch to a SPARCstation where the driver software resides.

- Configuration software for the ATM hosts
  The ForeRunner NICs are shipped with an SNMP agent that supports the Fore MIB. The Fore MIB provides ATM layer statistics but no network layer parameters. For the IP/VC mapping a command-line tool (`atmarp`) is enclosed. A similar tool (`pvcarp`) is enclosed with Interphase NICs.

- Configuration software for the ATM switch
  The LattisCell switch can be configured by a *connection management system* (CMS). It is installed on the SPARCstation which is connected to the switch at the 10BaseT-port. The CMS includes an SNMP-agent and sophisticated graphical tools for all kinds of statistics. As with the Fore MIB, the SynOptics MIB is read-only and was not designed for configuration purposes. VCs must be configured with a tool (`lcell_pvcbatch`). In addition, a *network management application* (NMA) allows integration into the SunNet Manager platform.

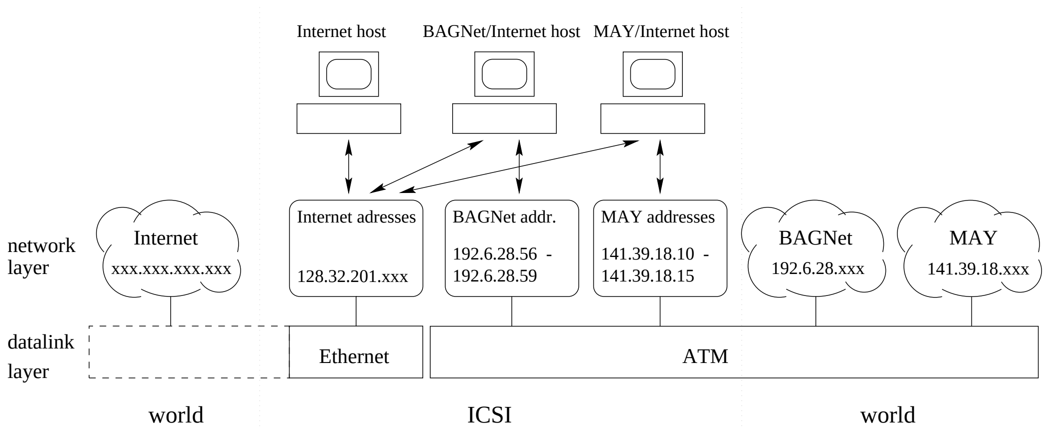


Figure 3.1: IP networks at ICSI

IP over ATM at ICSI

The outstanding role of IP in today's computer networks led to a quick development for both IP-based multi-media applications and solutions for overlaying IP on ATM networks. Although it hides ATM's superior feature of a guaranteed QoS, and packet-routing can decrease traffic throughput, IP can still benefit from ATM's multi-megabit transmission rates. ICSI is involved in two wide-are ATM pilots. Both implement IP at the network layer. Hence, together with the Internet, ICSI is part of three distinct IP-networks (Fig. 3.1). Tab. 3.1 summarizes essential concepts for IP over ATM.

The ATM environment at ICSI is PVC-based. At each host, local and remote destination IP addresses are mapped statically to VCs with one dedicated VC per destination. Hence, VC multiplexing could be used as packet encapsulation at the AAL layer. However, LLC/SNAP encapsulation as used with 'Classical IP over ATM' is used. An ATM connection management tool for ICSI's environment must provide a function to perform IP/VC configuration at the hosts. This requires a mapping scheme and the administration of IP addresses and VPI/VCI pairs.

Concept	**Description**
LANE [3]	A standard by the ATM Forum for emulating Ethernet or Token Ring LANs on ATM networks. LAN emulation (LANE) easily adapts to any traditional network layer protocols on ATM but hides its QoS features.
LLC/SNAP encapsulation [5]	Similar to traditional data link layer protocols a *Logical Link Control/Subnetwork Access Point* header identifies the upper layer protocol type. This allows for multiplexing different protocols on a single virtual connection.
VC Multiplexing [5]	An encapsulation method without packet type identifier. Different network layer protocols have to be multiplexed on different virtual connections since only one single protocol can be carried on a virtual connection. No packet type field is used since the type of protocol is defined implicitly at connection setup. VC multiplexing is used with LANE.
Classical IP over ATM [11]	In traditional shared-media LANs like Ethernet, *address resolution protocol requests* (ARP requests) are broadcasted on an IP subnet. ATM does not feature a subnet analogon and broadcasts are more complicated to implement. The concept of 'Classical IP over ATM' introduces the notion of *logical IP subnets* (LIS) with at least one ARP server per subnet. Nodes within a LIS address to the ARP server in order to query ATM addresses. Classical IP over ATM does not change the host requirements in terms of address resolution (hence 'classical'). LLC/SNAP encapsulation is used.
Manual configuration	Instead of using an address resolution protocol, maintaining ARP tables manually remains an option for small SVC-based networks. This solution does not require an ARP server, however, it is not scalable.
IP/PVC mapping	IP-over-ATM in PVC-based networks does not require address resolution in terms of an address-to-address mapping. Instead, IP addresses have to be mapped directly to a previously established VC. The ARP tables have to be maintained manually at each ATM network.
I-PNNI	The ATM Forum is developing an *integrated private* NNI (I-PNNI) since April 1996. The I-PNNI routing protocol extends the capability of PNNI routing to the IP protocol, i.e. it provides QoS support for IP applications.

Table 3.1: Concepts related to IP over ATM

The BAGNet project

Since 1994, the *Bay Area Gigabit Network* (BAGNet) has been connecting 15 institutions and companies in the San Francisco Bay Area [8]. The IP-over-ATM pilot reaches 60 hosts that are fully meshed via PVCs. BAGNet implements Classical IP over ATM (without ARP servers) on top of the mesh. 60 ARP table entries at the 4 hosts and 480 VC entries at the switch have to be configured for unicast connections to/from ICSI. For multicast traffic, additional entries are necessary. These entries are set up statically by configuration scripts at boot-time.

End-to-end connectivity is a critical issue in BAGNet. A special tool (`bagping`) checks all point-to-point connections periodically. The results are collected by email to generate a connectivity matrix that is displayed on the World-Wide-Web. The matrix was useful to detect connection failures. However, there is no tool at the local site that provides an integrated view of the configuration tables in order to track the failure. Moreover, keeping the configuration data in separate scripts bears the risk of address conflicts with other PVCs.

The MAY project

The *Multimedia Applications on Intercontinental Highway* prohect (MAY) was started in 1995. MAY was funded by a consortium of eleven research institutions, companies, and telecommunications operators in the USA and Europe. There are currently five sites with ATM hosts connected to MAY. Its subject are multi-media applications over long distances. For MAY, host-to-host and loop-back connections at various speeds have to be established. Only a small number of intercontinental PVCs can be used by a site at a time[1]. This implies frequent reconfiguration tasks. A connection management tool should help minimize the preparation time for MAY sessions.

Local ATM experiments

In addition to the multi-institutional ATM trials, ICSI performs many local multimedia-experiments that require end-to-end connectivity between ATM hosts within the LAN. The management tool must help organize and coordinate both local and non-local connections.

3.2 Qualities and attributes

The connection management issues of ATM in general and the specific properties of ICSI's ATM infrastructure imply a number of requirements for the software tool to be developed. This section summarizes the qualities and attributes the tool has to show.

[1]currently 4 VCs are assigned to ICSI (VPI/VCI 0/272 - 0/275).

Flexibility
ICSI's ATM environment is subject to frequent configuration changes. The connection management tool has to be flexible enough to adapt to changes concerning

- device configuration
  Adding or removing ATM devices to the network or exchanging ATM NICs of hosts must not affect the usability of the tool. It must be adaptable to hardware changes as well as to operating system updates and new versions of configuration software.

- address configuration
  The tool must allow frequent changes of the local IP/VC mapping (e.g. due to the lack of VCs at the public UNI). It also must be able to renumber VC identifiers at the public UNI according to configuration changes of the adjacent operator. The tool has to comprehend when hosts get connected or disconnected at remote sites to offer a correct overview of potential destination addresses.

- connection configuration
  Currently ICSI's hardware environment allows only for connections with *unspecified bit-rate* (UBR), i.e. there is no guaranteed bandwidth per connection. However, a peak rate can be specified to limit the maximum throughput. Different test-scenarios require different bandwidths. The connection management tool has to take this into account.

Integration
ICSI's current connection management procedures show a lack of integration to many respects. The new tool should ease connection management tasks by a high degree of integration:

- data integration
  Connection management data concerning device and connection parameters should be kept consistent and iredundant. The access method should be uniform for different device classes (hosts, switches) and independent of the location of the device.

- functional integration
  The tool should integrate different tasks of connection management

like connection setup and setdown, updating configuration data and monitoring traffic. Currently these tasks require various tools that are dedicated to a certain device and function. Moreover, it should integrate inter-dependent configuration tasks to powerful functions.

- integration of the user interface
  A uniform look-and-feel and easy access to all functions is necessary to retrieve connection management information quickly. The tool should be used not only by a network administrator but by all ATM users (e.g. for fault management or traffic monitoring). Hence, a simple and comfortable user interface is needed.

Abstraction and transparency
The tool should - as far as possible - hide all attributes of the ATM environment that are unimportant with respect to connection management. Devices should be represented by their essential parameters (device types, names, addresses). Parameters like VC identifiers or the type of ATM NIC are not of interest from the users point of view. The tool, however, must be aware of these configuration details in order to access the devices appropriately. Hence, the tool must provide both a means to store and update detail information and a mechanism to retrieve this information automatically without prompting the user.

In addition, any local ATM device should be accessible transparently from any host of the LAN. In other words, the location where the tool is invoked must be independent from the location of the target devices. This should be true not only for ICSI but also for remote ATM sites that should have insight to ICSI's ATM network (e.g. MAY participants in another time zone). Remote accessibility can be helpful to verify the right setting without support from ICSI. This requirement, of course, gives rise to security questions. The tool must feature a mechanism for authentification and authorization.

The requirements outlined above have to be translated into a design concept. Although the tool is aimed at ICSI's environment, the intention is to develop a design that is as independent as possible from local specifics. The next chapters cover the design and the implementation of TOMCAD - a tool for configuration and management of ATM devices.

Chapter 4

Design of TOMCAD

TOMCAD meets the requirements outlined in the previous Chapter by a web-based system architecture. This Chapter introduces all system components and their internal and external interfaces. A core function of TOMCAD is the management of configuration data in a database. Section 2 considers TOMCAD from a data-oriented point of view and explains how ICSI's ATM environment is reflected by the database. In the last section, the connection management functions are described with respect to input and output parameters and their effect on the database.

4.1 System architecture

The architecture of TOMCAD is considered with regard to the concerned networks, system components and their internal and external interfaces.

4.1.1 Involved networks

Accessability from local hosts as well as from external sites is achieved by a web-based system architecture which is open to the Internet. The wide-area ATM network itself cannot be the carrier for remote access, since it is subject to configuration. The Internet, however, provides permanent connectivity to ICSI (Fig. 4.1). Another advantage of a web-based approach is that no specific software has to be provided at other sites. TOMCAD can be developed, installed and maintained independant from any client system.

A view to ICSI's ATM environment over long distances is useful especially

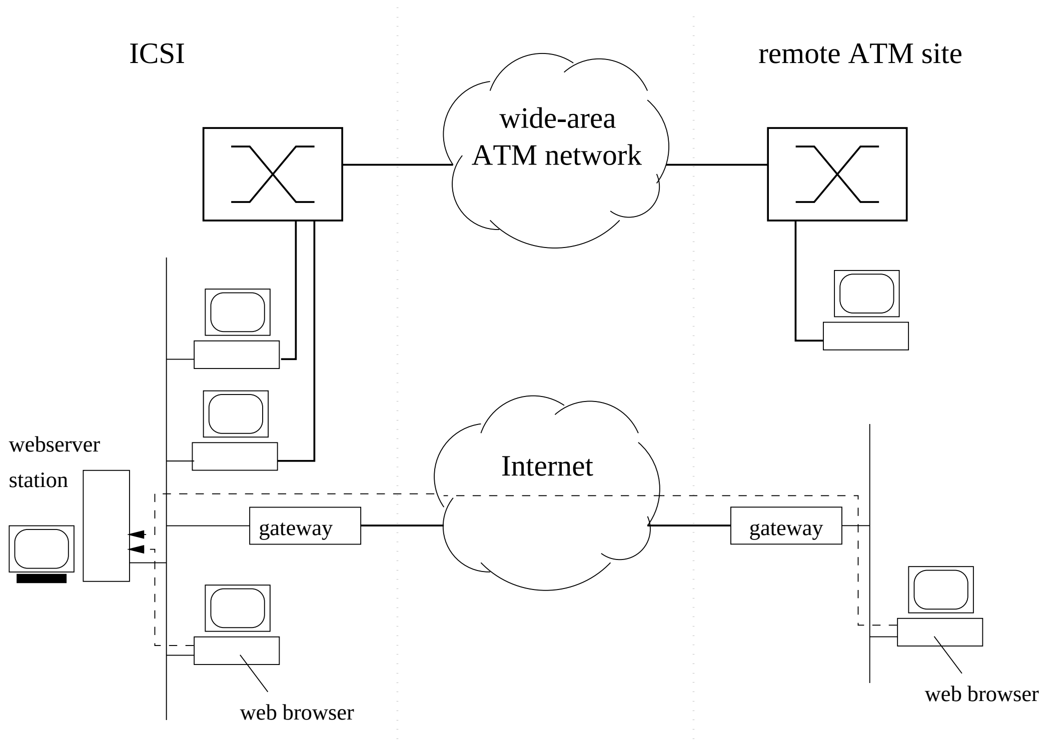


Figure 4.1: Remote access to ICSI's ATM environment via the Internet

for the MAY network. This network is often used for loop-back connections that allow for measurements of transmission delays or jitter occuring on intercontinental connections. Such experiments can be performed at one institution independant from the other site, assuming the loop-back is setup. A web-based tool can provide a means to establish or verify a loopback at ICSI's switch. This is appropriate, since the accessing party is loacted in a different timezone.

TOMCAD resides on a node in ICSI's IP-network which is part of the Internet. It addresses any local ATM device over this Ethernet-based IP-network. All ATM hosts have two network interface adapters that are connected to The Ethernet LAN and to the ATM network, respectively. Hence, ATM hosts have two different names for identification in the Internet or the ATM network. TOMCAD refers to them with their Internet hostname. The ATM switch is not an IP-node but is represented by its management station where the configuration software resides. TOMCAD communicates with this station to configure and monitor the switch (Fig. 4.2).

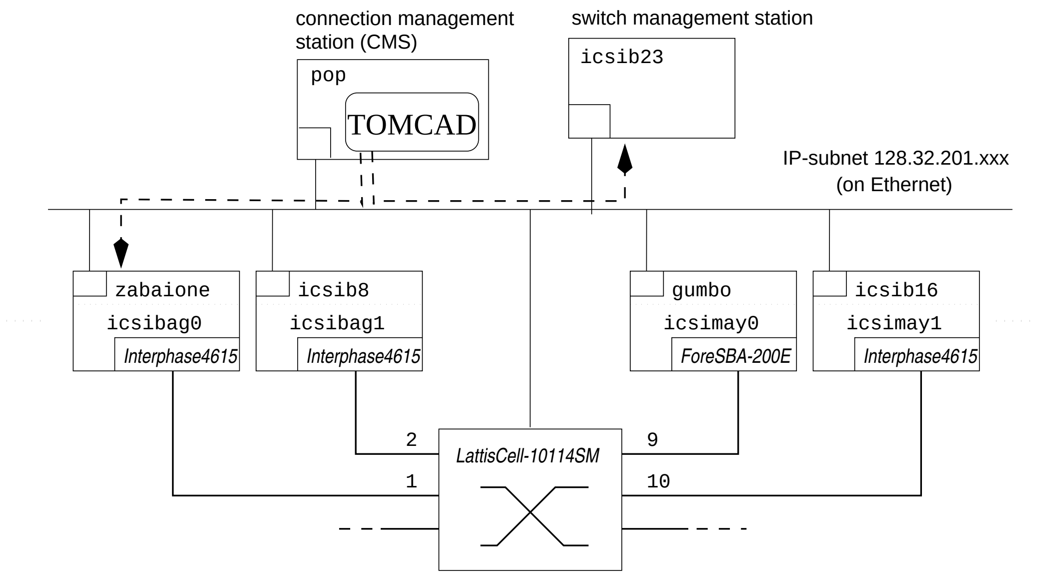


Figure 4.2: ATM device access by TOMCAD

4.1.2 System components

All TOMCAD components are located at the *connection management station* (CMS). Fig. 4.3 shows the components and their internal interfaces. Rounded rectengulars represent server processes that are permananently running. The other rectengulars represent code that is processed or invoked by another process. The external interfaces to the CMS are considered in the next section.

The components and their interaction with other components is described below:

- web server
  The web server is the core process of TOMCAD. It handles HTTP-requests in two different manners. Either by retrieving HTML-code that is stored at the local file-system, or indirectly through a CGI-binary that processes the request and generates HTML-code.

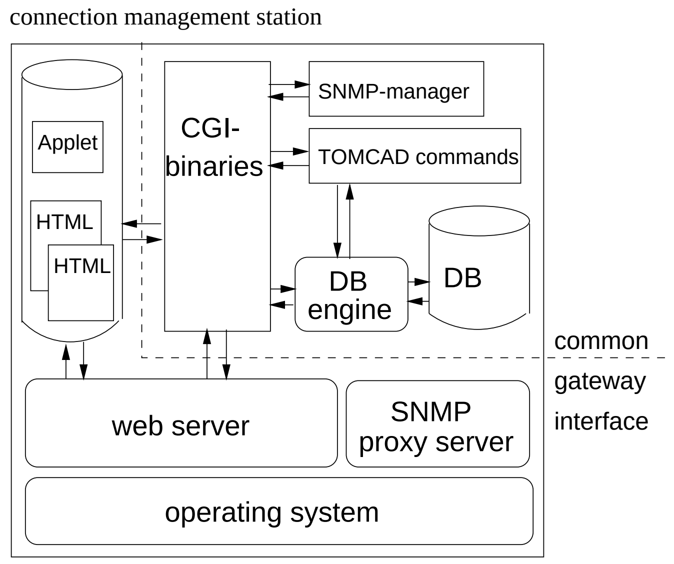


Figure 4.3: System components at the CMS

- CGI-binaries
  Every connection management task - except traffic monitoring - corresponds to a transaction that is performed by a CGI-binary and possibly other commands. When the web server receives a request that refers to a CGI-binary, it creates an environment defined by a set of parameters (including the user data sent with the HTTP-request) and executes the CGI-binary. The web server cannot control the execution. The CGI-binary performs its function depending on the environment variables. It does so by invoking other executables that perform subtasks. The HTML-coded output stream of the CGI-binary is passed to the web server. It is the result of the transaction.

- SNMP manager
  The SNMP-manager is invoked by CGI-binaries. It performs certain monitoring tasks and returns the retrieved raw data back to the CGI-process. The CGI-process, in turn, formats the data to HTML and passes it to the web browser.

- TOMCAD commands
  For complex connection management tasks it is useful to design specific executables that are independant from the CGI. CGI-binaries establish

environments for these executables, invoke them with appropriate arguments and postprocess their output-streams to generate HTML-code.

- The database engine
  The management database is an essential component of TOMCAD since almost every connection management task affects it. In most cases, it is a CGI-binary that performs the database operation. The TOMCAD commands can access it directly in order to perform operations on a set of entries. Hence, this data does not need to be provided by a CGI-binary via the process environment.

- HTML-files
  Various HTML files are part of the system. Having received an HTTP-request, the web server reads the demanded HTML-file and sends the content to the web browser. Since HTML-files are parsed by the web browser they can be considered as static data that does not need to be processed any further by the CMS. TOMCAD, however, makes extensive use of an extended syntax that allows for generating webpages dynamically (see Sec 5.5). Files that contain this extended syntax have to be compiled by a certain CGI-binary.

- Applets
  Some HTML-files include applet-code that is used for traffic management (Sec. 5.4). Applets are not executed at the CMS so they are sent out by the web server without further processing.

- SNMP proxy
  In contrast to all other software components at the CMS the SNMP proxy server has no internal interfaces. Its purpose is to relay traffic between external processes (see Sec 5.4).

4.1.3 External interfaces

Fig. 4.4 shows the interfaces between the CMS and external software components. These components reside on the client-station, ATM hosts and the switch management station.

At the client station a web browser provides an interface to the user and to TOMCAD. The CMS is the only node in the network the client-station communicates with. The web browser initiates management tasks and the

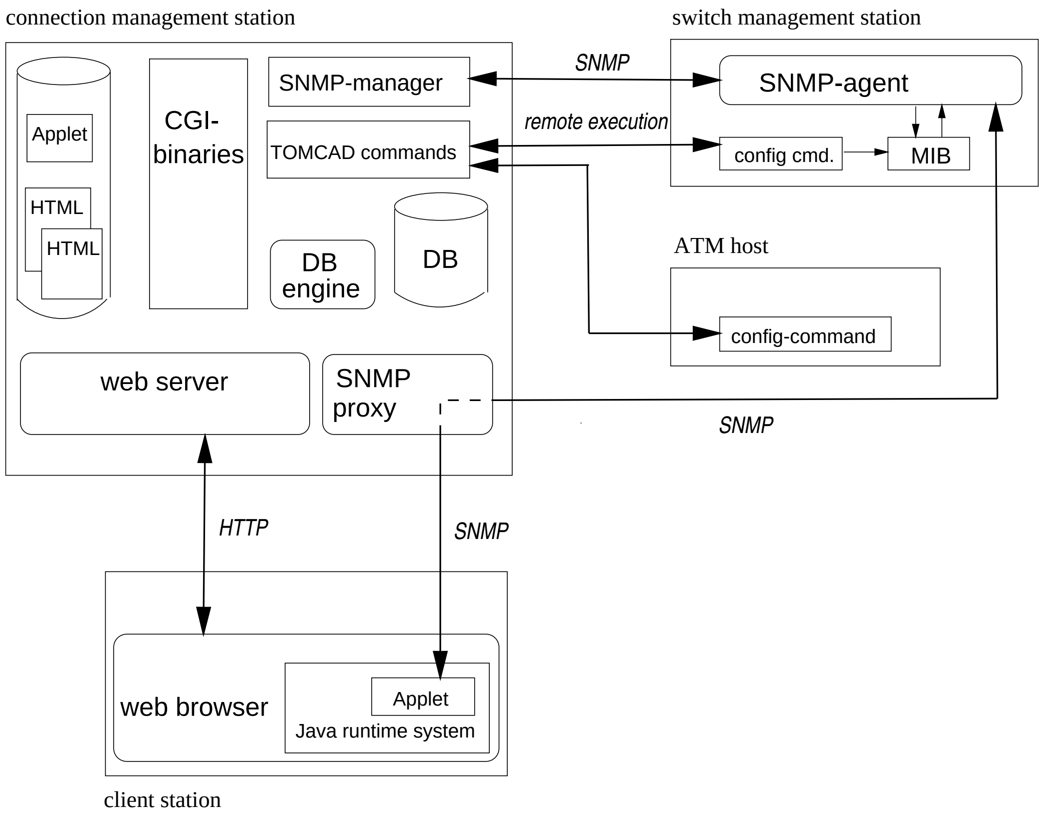


Figure 4.4: External interfaces of the CMS

web server returns the results - both via HTTP. For one special task, however, another protocol is used between these systems. For traffic monitoring, a Java Applet is executed at the client-station in order to continuously retrieve data from the switch management station. The SNMP traffic is relayed by the SNMP- proxy server in both directions. An SNMP-agent answers the requests at the switch managemet station. The agent is accessed not only by the proxy server but by another process at the CMS: the SNMP-manager, - invoked by a CGI-binary - queries the agent for status information. TOMCAD uses remote execution for tasks where HTTP and SNMP cannot be employed. Remote execution concerns hardware-specific configuration tools at the ATM hosts and the switch management station. At the CMS, special TOMCAD commands adapt to the input and output format of these tools.

4.2 The Management database

TOMCAD keeps all necessary configuration data in a central management database. This section describes how the database reflects the ATM environment. An *entity relationship model* (ER-model) is introduced that illustrates how ATM devices and connections are regarded by TOMCAD. The structure of the database tables is deduced from this ER-model. A description of the mapping scheme for VPI/VCI pairs to virtual connections concludes this section.

4.2.1 Configuration and monitoring data

TOMCAD must provide a database in order to store all management data in a centralized and structured way as demanded in Sec. 3.2. Management data can be categorized with respect to its purpose:

- system configuration data
  All parameters that characterize the installation of a device belong to this category. This includes hard- and software specific parameters (like ATM NIC type or operating system version) as well as addresses and names that identify a device. System configuration data is static. It changes only if new devices are added, if software is updated or if addresses and names change (e.g. for new project participants).

- connection related data
  This kind of data represents physical or logical connections. Physical connections refer to the arrangement of adjacent devices. Logical connections refer to the parameters that define a virtual connection. Logical connections can be enabled or disabled. Connection related data is more often subject to changes than system configuration data.

- status and traffic information
  This category comprises data which is the result of monitoring. This includes status parameters, configuration settings and the results of traffic measurements. Since this is volatile data, it is saved in the database only for some exceptions (see Sec. 4.3).

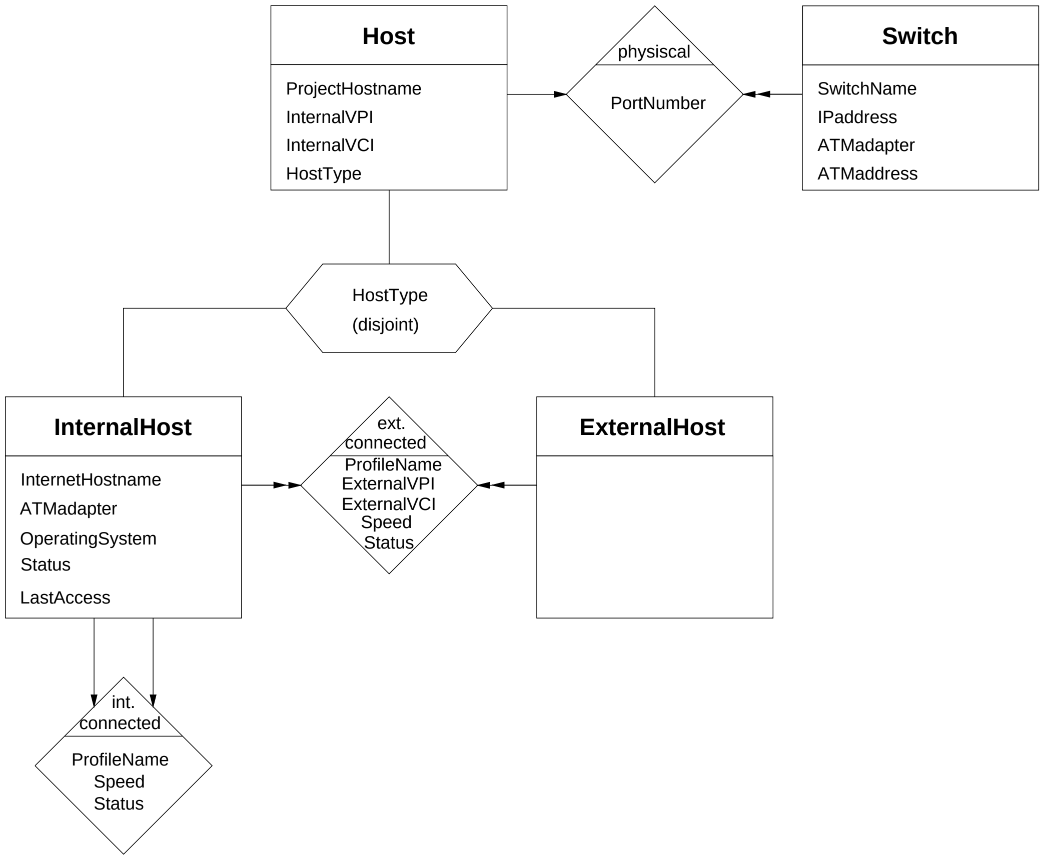


Figure 4.5: ER-model for connections and devices

4.2.2 An Entity-Relationship model

TOMCAD is restricted to management tasks at the UNI. The tool must handle local end-to-end connection as well as end-sections of non-local ones. The *entity-relationship-model* (ER-model)[1] introduced in this section shows how both kinds are taken into account (Fig. 4.5). A connection - in terms of TOMCAD - refers to a host-switch-host concatenation. Both hosts are physically or logically adjacent to the switch, i.e. they are both part of the LAN and directly attached to the switch or one host is located at a remote site, i.e. indirectly connected. Consequently, TOMCAD distinguishes between internal and external hosts and between internal and external end-

[1]The ER-model was invented by Chen [4].

to-end connections. Both kinds of hosts are connected to a certain port of the switch. For external hosts, this is the port to the public UNI.

The ER-model generalizes internal hosts and external hosts to reflect that they are treated the same way with respect to their connection to switches. Internal hosts are defined by a number of additional parameters since they are subject to configuration. Internal and external connections have different attributes as well for reasons explained in the next section.

Loop-back connections have to be considered in a special way to fit into this concept. Although they end at two internal hosts, loop-backs affect the public-UNI and are, hence, external. TOMCAD maps a loop-back on two distinct connections to two different external hosts. These hosts are actually aliases for the internal destination hosts. This solution helps to keep the connection model simple.

4.2.3 Assignment of VPI/VCI pairs

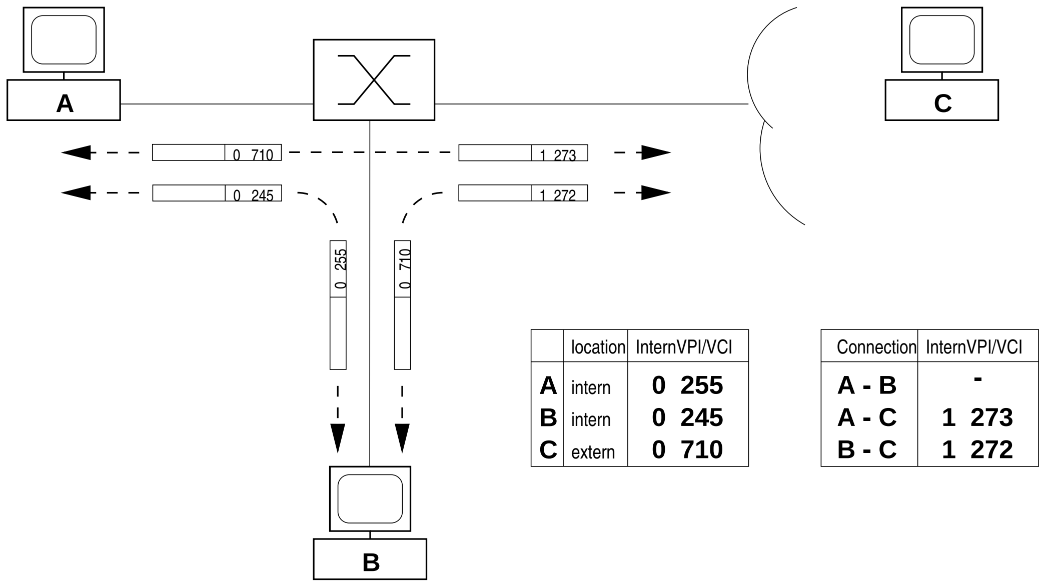
A virtual connection is a concatenation of virtual channels that are identified by pairs of VPIs and VCIs (see Sec. 2.1.2). TOMCAD has to specify VPI/VCI pairs in order to setup PVCs. For the public UNI, the values have to be selected from a fixed set of pairs according to the agreement with the operator of the public network. At the private UNI, the identifiers can be chosen arbitrarily. To ensure that VPI/VCI pairs are used uniquely on one ATM link, a mapping scheme is used by TOMCAD that determines the selection.

Each host has two attributes named `InternalVPI` and `InternalVCI`. This value pair has no meaning to the host it is assigned to. All other hosts, however, use this pair for VCs that are directed to this host. In other words, an internal host sends cells on VC (x/y) to the host with `InternalVPI=x` and `InternalVCI=y` (Fig. 4.6).

This scheme is symmetric for internal connections but can not be used for external PVCs, since the number of external hosts may exceed the number of available VCs. Hence, VPI/VCI pairs associated with the connection (not with a host) are used at the public UNI (`ExternalVPI, ExternalVCI`).

4.2.4 Database tables

The entities and relationships in the ER-model are reflected by tables in the database. Additional tables store monitoring data temporarily or contain



	location	InternVPI/VCI
A	intern	0 255
B	intern	0 245
C	extern	0 710

Connection	InternVPI/VCI
A - B	-
A - C	1 273
B - C	1 272

Figure 4.6: Mapping scheme for VPI/VCI pairs

the value range for certain attributes (e.g. ATM NICs or operating systems). The tables and their fields are depicted in Fig. 4.7.

The tables ATMBOARDS, OPSYS, and PROFILES each store a set of identifiers that are needed when it comes to database updates by the user. They provide the contents for selection lists or pull-down menus at the user interface. The PROFILES table keeps possible values for the `ProfileName` field of the CONNECTIONS table. Profiles label a set of connections in order to group them. For example, a profile might refer to connections with a certain speed or to connections that fully-mesh several ATM hosts.

HOSTS, RHOSTS and SWITCHES reflect the entities InternalHost, ExternalHost and Switch in the ER-model. The relationship physical is included in the HOSTS table. The fields `ProjectHostname` and `InternetHostname` identify an ATM host within the IP-over-ATM network and the Internet, respectively. `ATMadapter` indicates the ATM hardware type of hosts and switches; `ATMaddress` is needed to configure the connection table of a switch. The relationships int.connected and ext.connected are represented by the CONNECTIONS table. The `Location` tag marks each connection appropriately. For internal connections the `ExternalVPI` and `ExternalVCI` flag is null. Whether a connection is set up (active) is indicated by the `Status` field. The fields

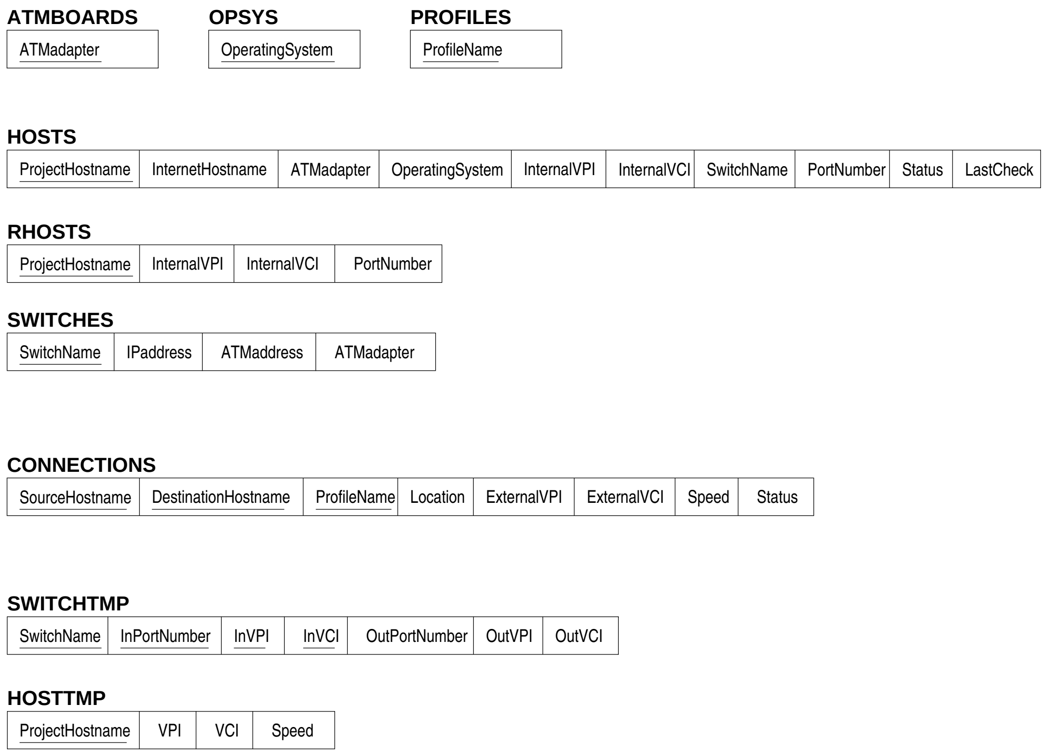

ATMBOARDS
ATMadapter
OPSYS
OperatingSystem
PROFILES
ProfileName
HOSTS
ProjectHostname InternetHostname ATMadapter OperatingSystem InternalVPI InternalVCI SwitchName PortNumber Status LastCheck
RHOSTS
ProjectHostname InternalVPI InternalVCI PortNumber
SWITCHES
SwitchName IPaddress ATMaddress ATMadapter
CONNECTIONS
SourceHostname DestinationHostname ProfileName Location ExternalVPI ExternalVCI Speed Status
SWITCHTMP
SwitchName InPortNumber InVPI InVCI OutPortNumber OutVPI OutVCI
HOSTTMP
ProjectHostname VPI VCI Speed

Figure 4.7: Database tables and fields

`LastCheck` and `Status` in the HOSTS table are updated when TOMCAD tries to access a host for configuration or monitoring purposes.

SWITCHTMP and HOSTTMP are tables that cache connection tables and ARP tables of hosts and switches, respectively.

4.3 Connection management functions

While the previous section provided a data-oriented view, this section covers the functionality of TOMCAD. All functions can be grouped in four classes with respect to their purpose. These classes are described in the following:

management of configuration data

All configuration data is stored in the management database. Some data reflects current settings in the ATM network, e.g. values of `Status` fields or the contents of SWITCHTMP and HOSTTMP. These values are set by TOMCAD automatically as the result of a configuration or monitoring task. All other data, however, is static with respect to configuration and monitoring; it has to be administered by the user in order to keep the database consistent with the network. Hence, TOMCAD features an interface that allows for inserting, updating an deleting entries. The user can access the database in two ways:

- by predefined database operations
- by a query language monitor

The first method is for adding, deleting or updating one record at a time in a user-friendly form-based manner. The second method requires knowledge about the table structure; it allows unrestricted access to the database. This interface is rather used for maintenance, e.g to check the consistency. In addition, a backup and restore function enables the user to save the current setting of the network.

setting and resetting connections

The main purpose of the configuration database is to provide information that can be automatically retrieved by TOMCAD in order to establish connections. TOMCAD's scope is limited to the UNI, hence at most two local hosts and one switch have to be linked. The input parameter of this function is a pre-defined connection (i.e. a record of the CONNECTIONS table). The whole configuration task comprises several subtasks:

1. Retrieve all necessary data about the devices from the database.
2. Remote-access the first host and update the ARP table.
3. Remote-access the switch and update the connection table.
4. If the second host is a local host, remote-access it and update the ARP table.
5. Update the `Status` field in the CONNECTIONS table appropriately.
6. Report the results to the user.

TOMCAD allows for establishing or resetting an entire profile at once. The procedure above is performed for all connection of a profile and a summary of all tasks is reported to the user.

monitoring device parameters

TOMCAD provides functions to monitor device parameters that are of interest with regard to connection management. This is at first the ARP table for ATM hosts and the connection table for switches. The user has to supply the name of the device. TOMCAD retrieves device-specific information from the database in order to access the devices appropriately. The raw monitoring data is formated for caching in the SWITCHTMP and HOSTTMP tables. These tables grant quick access but have to be updated periodically.

For switches, several additional parameters can be monitored. These are event time-stamps, cell-counters, and status indicators for the switch as a whole and per port.

Another simple function allows for verifying whether a virtual connection is established between two hosts. The user selects two hostnames of interest. TOMCAD, in turn, remote-executes a `ping`-command which contacts the other host. If the command fails, the CONNECTIONS table is updated, i.e. the `Status` flag is reset for the the appropriate entry. A fault can be traced by verifying the ARP-tables and connection tables of each concerned device.

continuous traffic monitoring

TOMCAD provides basic traffic monitoring for one virtual connection at a time (see 5.4). The measured parameter is the payload throughput on that virtual connection. TOMCAD monitors counters at the local switch to calculate the ratio. The user can define the time space and the interval of the measurements. The browser displays the graph that reflects the dynamic changes during the past period. The graph is updated automatically. The monitoring process goes on until the user decides to interrupt it.

Chapter 5

Implementation of TOMCAD

This chapter covers certain aspects of the implementation. The first section considers the general procedure of calling a function and receiving the results. The second section explains all elements of the user interface in detail. Dependencies between executables at the CGI are described in section 3. Section 4 covers the the Java/SNMP-based implementation of traffic monitoring. A consideration of the MiniSQL database concludes this chapter.

5.1 Selecting functions and arguments

TOMCAD makes extensive use of HTML-forms to prompt for input. Only few operations can be performed by a single browser-server roundtrip. These are viewing operations on the management database and forward/backward-browsing (Sec. 5.2.1). All other operations require to go through a sequence of forms until all necessary information is supplied. Usually two browser-server roundtrips are necessary to accomplish an operation. Fig. 5.1 shows the general procedure how an operation is selected and invoked.

The procedure includes the following steps:

1. The user selects function `F` by clicking the referring link in order to perform a TOMCAD operation.

2. The browser requests the corresponding web page from the web server.

3. The web server invokes a special CGI-binary to generate the web page.

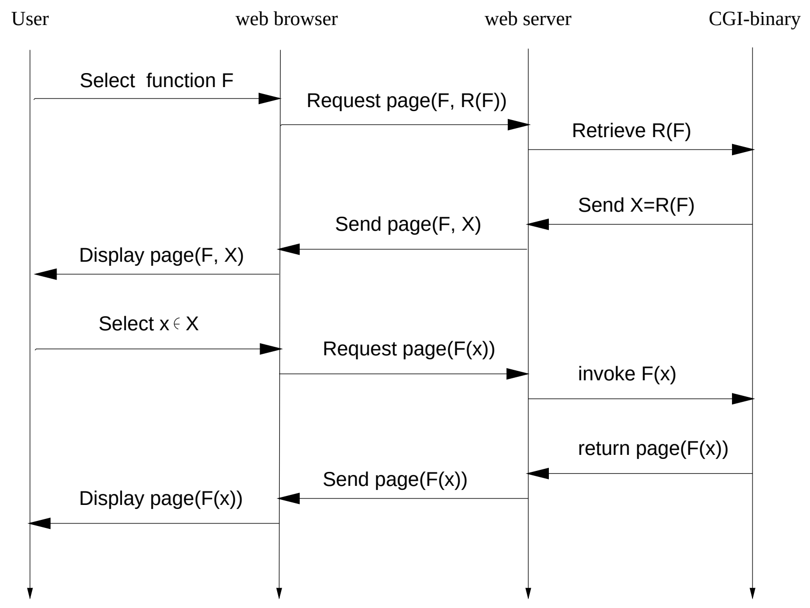


Figure 5.1: Selecting and invoking an operation

4. The CGI-binary performs a database query to retrieve the data applicable on `F` (`X=R(F)`) and formats it in HTML code.

5. The web server receives the output generated by the CGI-binary.

6. The web server, in turn, sends the page back to the browser.

7. The browser parses and displays the webpage. It contains a form with selection lists and pull-down menus that reflects `X`.

8. The user selects the objects (`x`) and sumbits the form.

9. The web browser passes the arguments to the server and requests the refering webpage.

10. The web server passes `x` to the CGI-binary that represents `F`. The CGI-binary generates a web page that contains the results of `F(x)`.

11. The web browser displays the results.

Few operations do not match this procedure. These are database operations that are fully specified by the hyperlink (e.g. viewing an entire table) or operations that allow for pre-selecting a subrange of `F` before the arguments are chosen. Such operations require an additional browser-server roundtrip. Pre-selections can be useful to avoid huge selection lists.

5.2 The user interface

An HTML extension, called frames, invented by Netscape Communications, Corp., allows to divide up a browser window in several rectengular areas. Each frame contains a distinct web page. Hyperlinks within one frame can refer to another frame as the target area. This allows for implementing areas for control and for input and output. TOMCAD's user interface employs frames to divide the browser window into two parts: the control panel and the *input output window* (IO window) (Fig. 5.2).

5.2.1 The control panel

The control panel is similar to the menu bar of GUI applications. Its purpose is to display the main functionality of the application and to provide a means for selecting it. Its layout is static and independent of the current context. It is always visible since every link refers to the other frame which contains the IO window. In other words, clicking a link in the control panel causes the IO window to be loaded with another web page while the panel on the left remains unchanged. All function items (hyperlinks) are displayed at the same time and are directly accessible, i.e. there is no analogon to pull-down menus. As a tradeoff for this approach smaller fonts and more space had to be used for the control panel . The items are grouped by context into five sections described below.

The Database Access Array
The database access array in the upper part consists of links to perform basic database operations like viewing, adding, deleting or updating entries. There is one column for each operation and one row for each maintained entity. For example, to update table OPSYS, a user would click the link `UPD` in the row with the header `OPSYS`. To display an entire table, the row header has to be clicked (e.g. the `OPSYS` link). For adding, deleting and updating operations (`ADD`, `DEL`, `UPD`), the user has to proceed through a sequence of forms before the database operation is finally performed (see 5.1). The user is asked to

select a single database entry (deleting, updating) and to fill in all parameter values (adding, updating). After submitting the last form to the server, messages about the transactions are displayed in a terminal-like mode. To verify the results, the table has to be selected for viewing as described above.

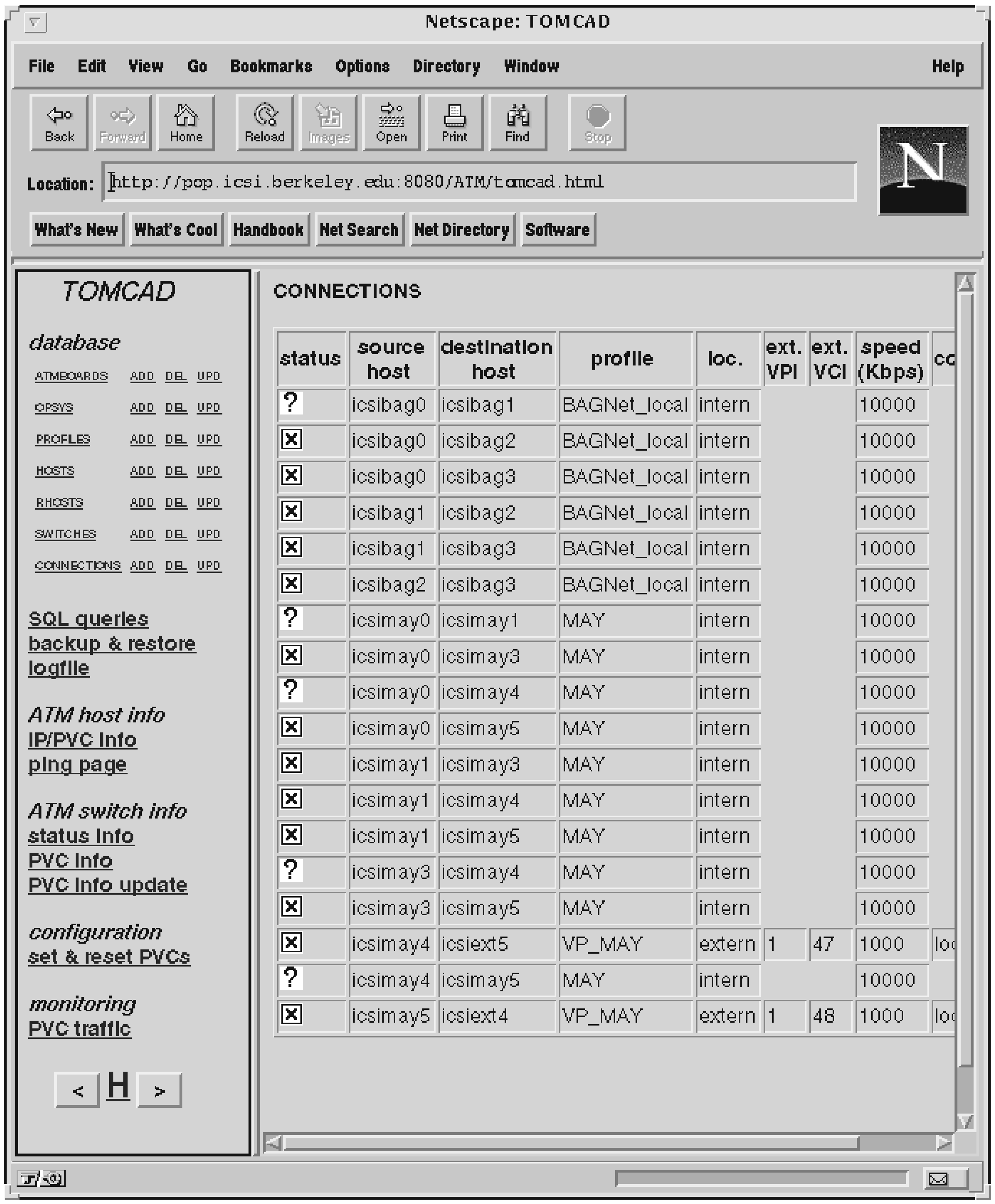
`SQL-Queries`
To perform database operations not offered by the database access array, the hyperlink `SQL queries` has to be clicked. It refers to a form with an input field for submitting MiniSQL-statements to the database engine (see Sec. 5.5. In response, the query results are displayed in plain text mode.

`backup & restore`
The `backup & restore` hyperlink addresses a form to specify a file the database should be read in from (restore) or written to (backup). Submitting the form invokes the appropriate command at the server. For safety reasons, a backup precedes each restore operation. Messages are reported in the IO window.

`logfile`
The logfile is affected by two tasks that concern the database: `backup & restore` and `VC update`. The `logfile` link refers to a text file where the time stamps of these tasks are recorded. The file keeps track of the filenames of all backup/restore operations and stores a time stamp when the SWITCHTMP table has been updated.

`IP/PVC Info`
To retrieve the IP/PVC configuration of an ATM host the `IP/PVC info` hyperlink has to be clicked. In response, the monitoring data is displayed as plain text. As an option, the HOSTTMP table can be viewed to get an HTML-formatted image.

`ping page`
This hyperlink is for verifying the reachability of an ATM host or switch. Moreover, the user can check whether a PVC between two ATM hosts is established or not. A web page containg three forms lets the user choose the host or the connection of interest. The results are displayed in the IO window and the status flags in the configuration table are updated appropriately.



CONNECTIONS

status	source host	destination host	profile	loc.	ext. VPI	ext. VCI	speed (Kbps)	co
?	icsibag0	icsibag1	BAGNet_local	intern			10000	
☒	icsibag0	icsibag2	BAGNet_local	intern			10000	
☒	icsibag0	icsibag3	BAGNet_local	intern			10000	
☒	icsibag1	icsibag2	BAGNet_local	intern			10000	
☒	icsibag1	icsibag3	BAGNet_local	intern			10000	
☒	icsibag2	icsibag3	BAGNet_local	intern			10000	
?	icsimay0	icsimay1	MAY	intern			10000	
☒	icsimay0	icsimay3	MAY	intern			10000	
?	icsimay0	icsimay4	MAY	intern			10000	
☒	icsimay0	icsimay5	MAY	intern			10000	
☒	icsimay1	icsimay3	MAY	intern			10000	
☒	icsimay1	icsimay4	MAY	intern			10000	
☒	icsimay1	icsimay5	MAY	intern			10000	
?	icsimay3	icsimay4	MAY	intern			10000	
☒	icsimay3	icsimay5	MAY	intern			10000	
☒	icsimay4	icsiext5	VP_MAY	extern	1	47	1000	lo
?	icsimay4	icsimay5	MAY	intern			10000	
☒	icsimay5	icsiext4	VP_MAY	extern	1	48	1000	lo

Figure 5.2: The TOMCAD user interface

`status info`
The `status info` item is for monitoring switch parameters. A form prompts the user to select a switch and to choose between general or port-specific parameters. The resulting web page displays the values (power status, up-time, cell counters etc.) in an HTML-table.

`PVC info`
The `PVC Info` operation for viewing an image of the connection table of an ATM switch. Since the number of entries in this table may be very high ($>$ 1000) the information is not retrieved on the fly but from the SWITCHTMP table that caches all entries. The user first has to specify the switchname and at least one port. If two ports are specified, only PVCs that run through these ports are displayed. Since point-to-point PVCs do not have a designated direction, the source and destination can be used interchangeably. The output is formated as an HTML-table.

`PVC info update`
This function updates the cache that keeps the connection table for a certain switch. The user has to select a switchname and submit the form to the CMS where the appropriate commands are invoked. The received results are parsed and stored in the SWITCHTMP table. This process sequence is executed in background at the CMS because retrieving big tables can last several minutes. During this time the http connection between the browser and the http-server at the CMS remains open, i.e. the browser keeps on waiting for the http response to finish. The reason for this is that the CGI-binary that invoked the background process will not finish until all its subprocesses have terminated. However, instead of waiting the user can click other links and interrupt the http-transfer. This does not affect the update process. A lock is implemented to ensure that calling `PVC info update` while an update is already running will not start another background process. The user can check the `logfile` whether the update has finished and call `PVC info` to view the results. An update should be performed after each backup because the cache table is deleted before the database is dumped.

`set & reset`
This link represents operations for connection setup and setdown. The user can select a single connection or a profile and clicks one of the submit but-

tons labeled `SET` or `RESET`. No further data has to be supplied by the user since TOMCAD retrieves all necessary parameters from the database. Each subtask performed by TOMCAD generates detailed output messages to keep the user informed about the state of the operation. A setup/setdown operation is considered to be successful if all affected devices have been configured successfully. As a last step, the status flag in the CONNECTIONS table is set to active or inactive, respectively. All messages are displayed in the IO window in a terminal-like mode (Fig. 5.5). A PVC update should be performed after every set/reset operation in order to update the cache tables.

`PVC traffic`
To access TOMCAD's traffic monitoring function `PVC traffic` must be clicked. The user is prompted to select the PVC he/she wants to monitor. In response, a web page is downloaded that contains two applets. Each applet displays the througput ratio for one direction of the PVC as a graph. The graphs are updated automatically without further user interaction (see Sec. 5.4).

Control Buttons
The last row of the panel shows two control buttons and a link labeled `H`. The link addresses the TOMCAD startup page that shows some information about the tool and its author. The control buttons are linked to two JavaScript-functions for browsing back and forth in the history stack of the browser. This can be useful to redisplay the messages of a previous operation. It should not be used to review database tables after a transaction, because the cached web page does not reflect the changes.

5.2.2 The Input/Output Window

The IO window is similar to the working area of traditional GUI-based applications. It is an input area where the user is prompted to specify the desired operations in detail and it is the output area where results and status messages are displayed. Output can be categorized in four classes by its format:

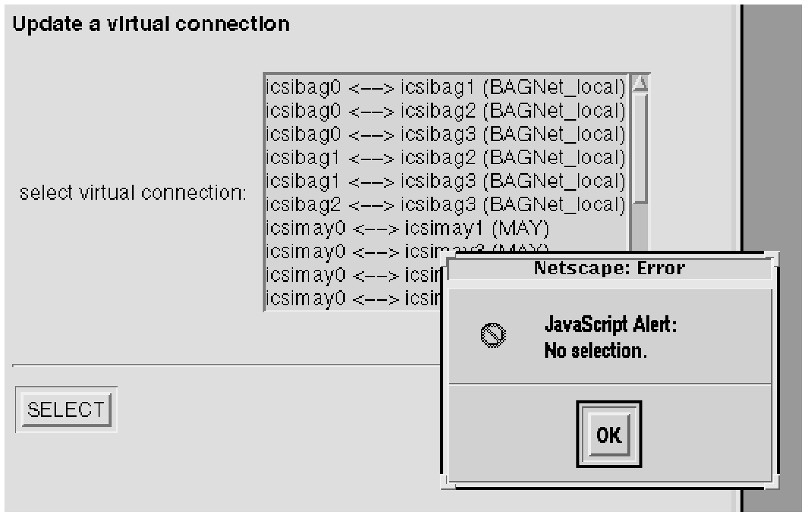
- plain text, used for status messages
- HTML-tables, used to display database contents and monitored device parameters
- HTML-forms, used for all kinds of user input (selecting items, inserting values)
- applets, used to dynamically display monitoring data
- pop-up dialog-boxes, used to login to TOMCAD and to notify the user about incorrect input data in forms.

User input can be categorized as follows:

- clicking a hyperlink, to select an operation
- filling out and submitting forms
  - to select data
  - to insert data
- to confirm a dialog box or to login to TOMCAD

TOMCAD makes extensive use of forms whenever a set of parameter values has to be entered. The following example shows how forms are used for updating a connection entry in the database. Fig. 5.3 and Fig. 5.4 display the IO window containing the corresponding entries.

The first form provides a selection-box with all connections. A connection is represented unambiguously by its source hostname, its destination hostname and its profile name. When the user submits the form by pressing the `SELECT`-button the browser checks whether an item has been selected or not. If this is not the case, the browser prevents the submission and generates


Update a virtual connection

select virtual connection:

icsibag0 <--> icsibag1 (BAGNet_local)
icsibag0 <--> icsibag2 (BAGNet_local)
icsibag0 <--> icsibag3 (BAGNet_local)
icsibag1 <--> icsibag2 (BAGNet_local)
icsibag1 <--> icsibag3 (BAGNet_local)
icsibag2 <--> icsibag3 (BAGNet_local)
icsimay0 <--> icsimay1 (MAY)
icsimay0 <--> icsi
icsimay0 <--> icsi
icsimay0 <--> icsi

Netscape: Error

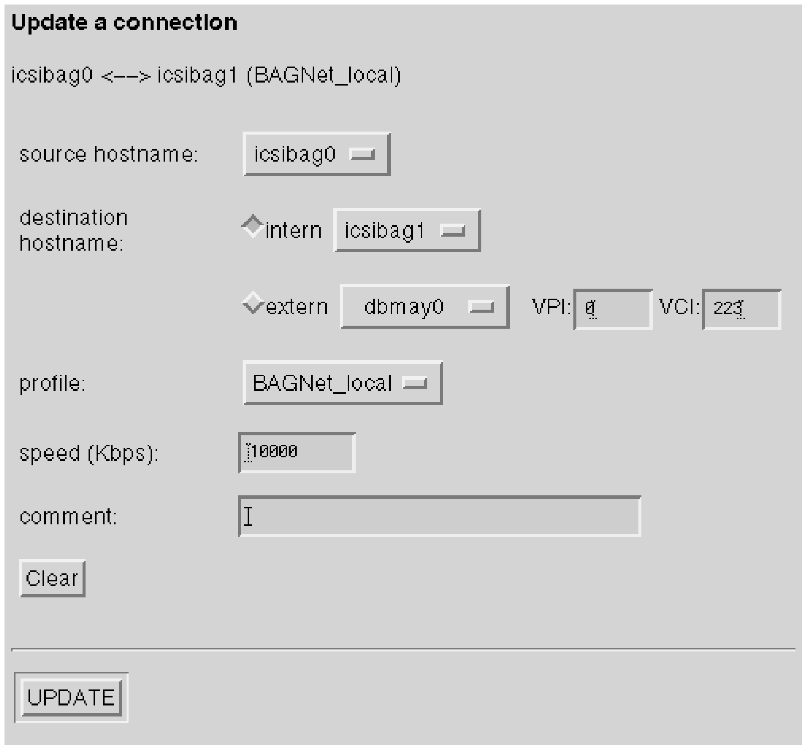
JavaScript Alert:
No selection.

Figure 5.3: Form submission denied

dialog-box to notify the user about the incomplete form. The user has to confirm the pop-up message and has to correct the problem (i.e. highlight an item in the selection box). If the submission succeeds, the web server responds with another form. The text slots and selection lists are preset to the current parameter values (Fig. 5.4). The user overwrites the values he/she wants to change and submits the form. Again, the browser checks the form contents and denies the submission if necessary.

Once the web server has received the new record, it starts a CGI-binariy that updates the appropriate database tables. Messages generated during the execution are sent back to the browser. Warning messages notify the user that similar data records might have caused a conflict while trying to overwrite the old record.

CGI-binaries generate messages continuously during their execution. Those that affect the management database have a quick response time compared to binaries for configuration tasks. The reason is that database commands are executed on the CMS while configuration tasks always include network communication with remote nodes. Setting up a connection that affects three nodes, for example, takes 10 to 20 seconds to finish. When several connections are to be set (batch task), the waiting time multiplies accordingly. To keep the user informed about the ongoing process at the CMS, the CGI-binaries continuously generate messages about each subtask

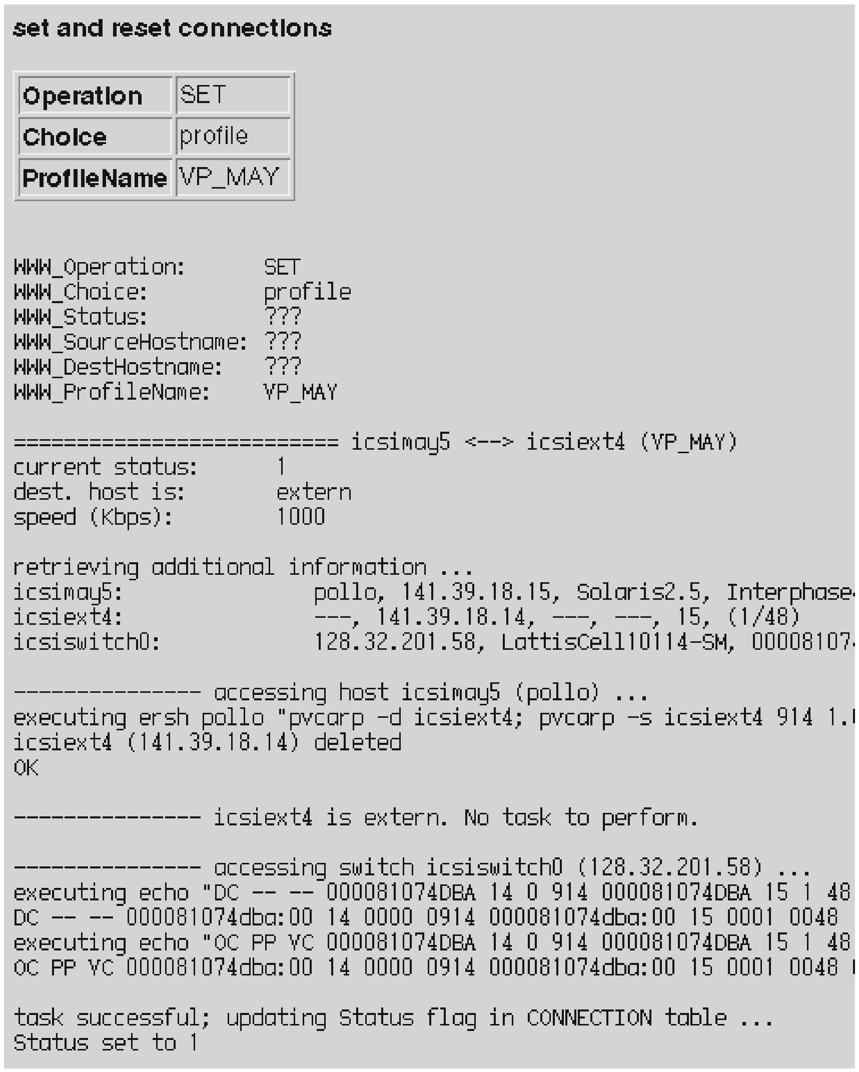

Update a connection

icsibag0 <--> icsibag1 (BAGNet_local)

source hostname:	icsibag0
destination hostname:	intern icsibag1
	extern dbmay0 VPI: 0 VCI: 223
profile:	BAGNet_local
speed (Kbps):	10000
comment:	

Figure 5.4: Updating a connection

(Fig. 5.5). Together, these messages form an activity log. Since there is no possibility to control a CGI-process during execution - except to abort it - it is important to have such a summary about the entire process. The user can analyze the log and rerun single tasks if necessary.



set and reset connections

Operation	SET
Choice	profile
ProfileName	VP_MAY

```
WWW_Operation:      SET
WWW_Choice:         profile
WWW_Status:         ???
WWW_SourceHostname: ???
WWW_DestHostname:   ???
WWW_ProfileName:    VP_MAY

========================= icsimay5 <--> icsiext4 (VP_MAY)
current status:      1
dest. host is:       extern
speed (Kbps):        1000

retrieving additional information ...
icsimay5:               pollo, 141.39.18.15, Solaris2.5, Interphase
icsiext4:               ---, 141.39.18.14, ---, ---, 15, (1/48)
icsiswitch0:            128.32.201.58, LattisCell10114-SM, 00008107

--------------- accessing host icsimay5 (pollo) ...
executing ersh pollo "pvcarp -d icsiext4; pvcarp -s icsiext4 914 1.
icsiext4 (141.39.18.14) deleted
OK

--------------- icsiext4 is extern. No task to perform.

--------------- accessing switch icsiswitch0 (128.32.201.58) ...
executing echo "DC -- -- 000081074DBA 14 0 914 000081074DBA 15 1 48
DC -- -- 000081074dba:00 14 0000 0914 000081074dba:00 15 0001 0048
executing echo "OC PP VC 000081074DBA 14 0 914 000081074DBA 15 1 48
OC PP VC 000081074dba:00 14 0000 0914 000081074dba:00 15 0001 0048

task successful; updating Status flag in CONNECTION table ...
Status set to 1
```

Figure 5.5: Output of a connection setup operation

5.3 Processes at the common gateway interface

Functionality on web servers is implemented at the common gate way interface. This interface specifies the environment variables of the CGI-process and the format of input and output to/from the web server. A CGI-binary can be written in any programming language. Scripting languages like the UNIX-shells or Perl are used most often.

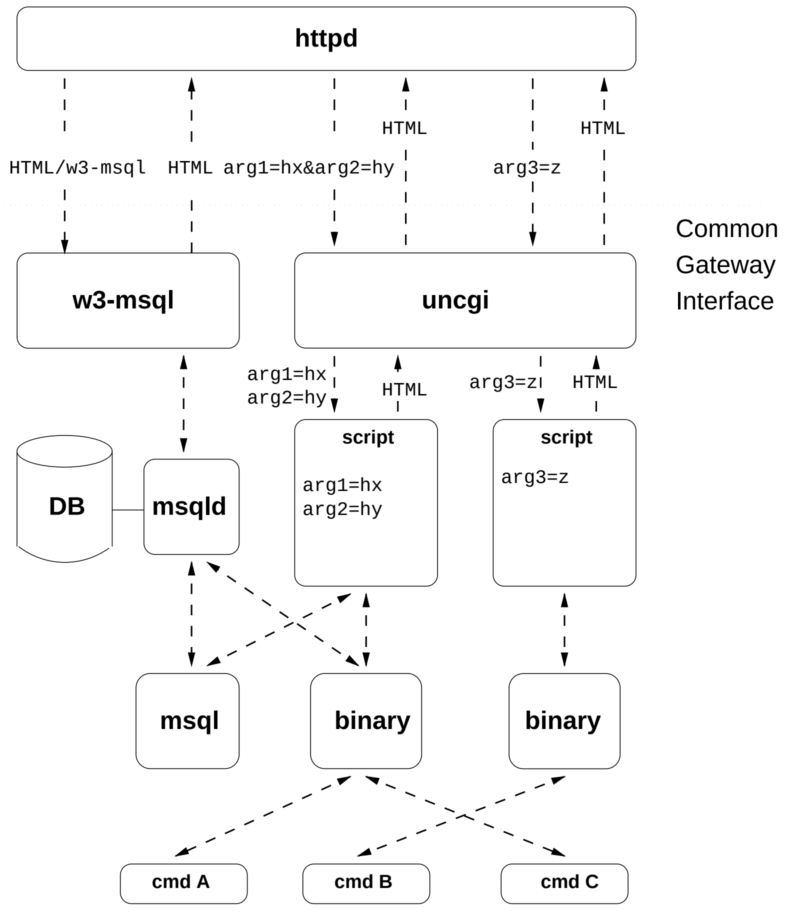


Figure 5.6: Process hierarchy at the CGI

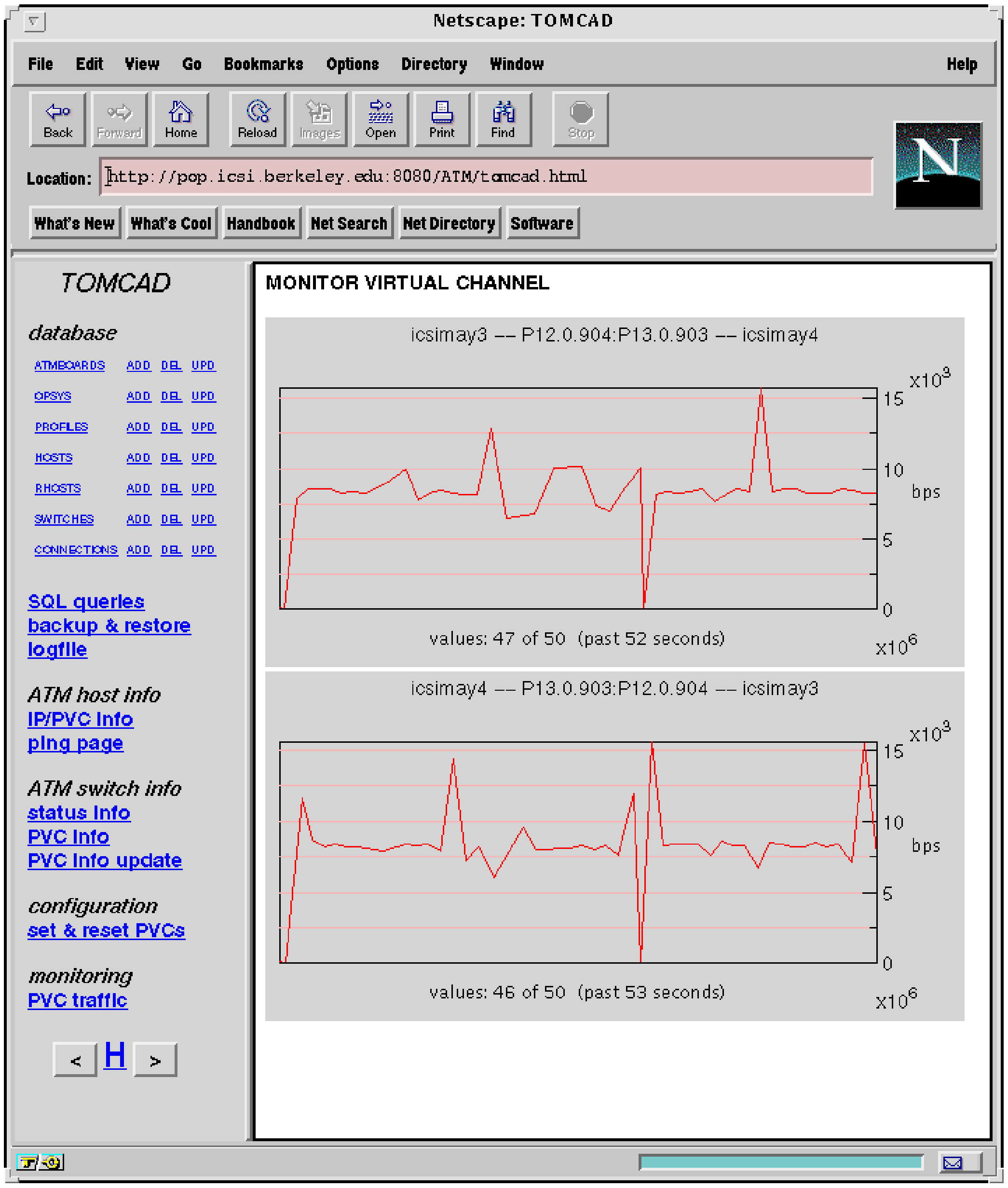
TOMCAD, however, uses both CGI-scripts and compiled executables (C-programs). The dependencies of executables at the CGI is depicted in Fig. 5.6. Form input is not passed directly to the CGI-program but through a filter (`uncgi`) that decodes the input. The CGI-script prepares an environment for the CGI-binary that performs the connection management function. The CGI-binary returns an output stream which is formatted to HTML by the script. The output of the script is, in turn, passed to the web server. Besides the binaries written for TOMCAD, other binaries are invoked by the scripts. The text-based SNMP-manager (`snmx`) that accesses the switch, for example, is invoked by a CGI-script [13]. Organizing code execution in two levels helps handling the complex tasks at the CGI.

5.4 Monitoring with Java applets

The traffic monitoring function of TOMCAD is fundamentally different from all other function because it cannot be performed by a CGI-process since its output always results in a single web page. If a CGI-binary continuously generates output over a longer period of time - e.g. in order to monitor a dynamic process - the web browser will accumulate this output until the CGI-binary has finished. This can cause huge web pages.

Netscape Communications defined a new content-type for web-pages that addresses this problem. This content-type specifies the web-page to be 'scrollable', and causes the web browser to present it in a terminal-like mode. A Netscape browser scrolls up the page automatically if necessary, and captures only a maximum of lines. This feature - although yet not supported by many browsers -, allows for continuous monitoring of single values. Further processing of these values e.g. for statistical purposes - is not possible.

Another HTML-enhancements by Netscape Communications for dynamic web pages is called client-pull. This concept enables the browser to automatically replace images within a web page after a specified time. The bowser (client) requests (pulls) the new image from the server. Slide-shows or simple animations can be implemented by this mechanism. However, since each image must be addressed by its predecessor one cannot created an image sequence dynamically. Hence, client-pull cannot be employed to implement traffic monitoring.



MONITOR VIRTUAL CHANNEL

icsimay3 -- P12.0.904:P13.0.903 -- icsimay4

values: 47 of 50 (past 52 seconds)

icsimay4 -- P13.0.903:P12.0.904 -- icsimay3

values: 46 of 50 (past 53 seconds)

Figure 5.7: Applets for traffic monitoring

Since HTTP and HTML do not support it adequately, the SNMP protocol was chosen for continuous traffic monitoring. The SNMP-agent of ICSI's switch provides counters that store the number of transmitted ATM cells for each PVC. An SNMP-manager process periodically queries the agent for the current value of a certain counter. It calculates the current throughput ratio and generates a graph that shows the recent development of the ratio (Fig. 5.7).

The manager process was implemented as a Java applet using the Java SNMP package by Advent Networks, Inc., and a Java Graph package[1]. As part of the initialization the applet process reads out parameters included in the HTML-page. These parameters specify the the switch management station, the PVC and the time interval between subsequent queries. The applet then starts a thread that periodically requests the counter level and together with a time stamp. The applet calculates the new throughput ratio based on the past two counter-levels and corresponding time stamps. A graph-object provided by the Graph-package includes methods to attach new value pairs, re-scale and redisplay the graph.

For security reasons the Java-runtime system restricts applets with respect to communication over the network. An applet can open a socket connection only to the host it was downloaded from. As a workaround, Advent offers an *SNMP applet server* (SAS). The SAS resides at the web server station and relays SNMP packets between applets and agents. TOMCAD employs an SAS at the CMS. From the programmers point of view the packets are send directly to the agent. The Java classes of the SNMP package abstract from the SAS server.

Although downloading the applet takes a considerable amount of time, the monitoring process, once started, shows very good performance that permits measurement intervals 0.5 seconds or less. The SNMP package provides a powerful programming interface for the development network management software.

[1]http://www-ipgg/llnl.gov/people/brookshaw/java/source.html

5.5 Database access methods

MiniSQL by Hughes Technologies, Inc. is a database product that was chosen for the implementation of TOMCAD's management database. MiniSQL is a light-weight database that can easily be installed on any UNIX system. It causes low processing load and requires few space in the file-system. The current version 1.0 only implements a subset of the *structured query language* (SQL). It is, however, sufficient for the management of configuration data which requires only simple operations on reasonable amounts of data. MiniSQL can be installed on any host of the LAN since other systems can connect via a pre-defined *transmission control protocol socket* (TCP-socket).

In case of TOMCAD, the database engine resides on the CMS since all accessing processes are executed there. As any other operation at the CMS, database operations must be performed by a CGI-process invoked by the web server. Different access methods are employed depending on the kind of operation. MiniSQL provides three different interfaces:

- the terminal monitor (`msql`)
  The terminal monitor `msql` is a command-line interface to MiniSQL. The user can submit MiniSQL-statements to query or manipulate the database. In response to a query a text table is displayed and the output of a database manipulation is a message about success or failure of the operation. TOMCAD uses the terminal monitor for adding, deleting or updating operations. Each operation - according to a field in the database access array (Sec. 5.2.1) - is implemented by a CGI-script that translates the received form input into a MiniSQL statement. It calls the terminal monitor to process the statement and reports the results to the user.

- The C programming library
  Complex TOMCAD operations, like connection setup, are implemented as an executable written in C. This executable includes function calls to the C programming library. In contrast to the terminal monitor and the web front-end query results can be browsed record-wise. For each record a monitoring or configuration function can be performed.

- the web-front-end (`w3-msql`)
  Hughes Technologies has developed a web-front-end for MiniSQL. This front-end is implemented as a CGI-binary that can parse HTML-pages

before they are send out by the web server. `w3-msql` recognizes a special syntax for MiniSQL statements which cannot be processed by web browsers. `w3-msql` resolves these statements, and substitutes the enclosed variables with the results of the database query. The HTML-page to be accessed is passed as an argument.

The following example illustrates how `w3-msql` is used to create a form that contains a selection-list (Fig. 5.8). Assuming the filename of the HTML-file is `enhanced.html` then requesting

```
http://pop.icsi.berkeley.edu/cgi-bin/w3-msql/enhanced.html
```

will invoke `w3-msql` which will compile `enhanced.html` that contains the following statements:

```
<! msql connect>
<! msql database TOMCAD>
<! msql query "select * from ATMBOARDS order by ATMadapter" result>

<FORM METHOD="POST" ACTION="/cgi-bin/ATM/atmboards/deleteatmboards">

 Select ATM board: <TD>
<select name="ATMadapter" size=10>
<! msql print_rows result "<OPTION value=̈@result.0¨>@result.0">
</select>
<hr>
<INPUT type="submit" value="DELETE">
</FORM>
```

The first three statements connect to the database and retrieve an ordered list of all ATM NICs. The `result`-handle refers to that list in the fourth statement. `w3-msql` resolves this statement with the appropriate values:

```
<FORM METHOD="POST" ACTION="/cgi-bin/uncgi/ATM/atmboards/deleteatmboards">

Select ATM board:
<select name="ATMadapter" size=10>
<OPTION value="ForeSBA-200E">ForeSBA-200E
<OPTION value="Interphase4615">Interphase4615
```

```
<OPTION value="LattisCell10114-SM">LattisCell10114-SM
</select>
<hr>

<INPUT type="submit" value="DELETE">
</FORM>
```

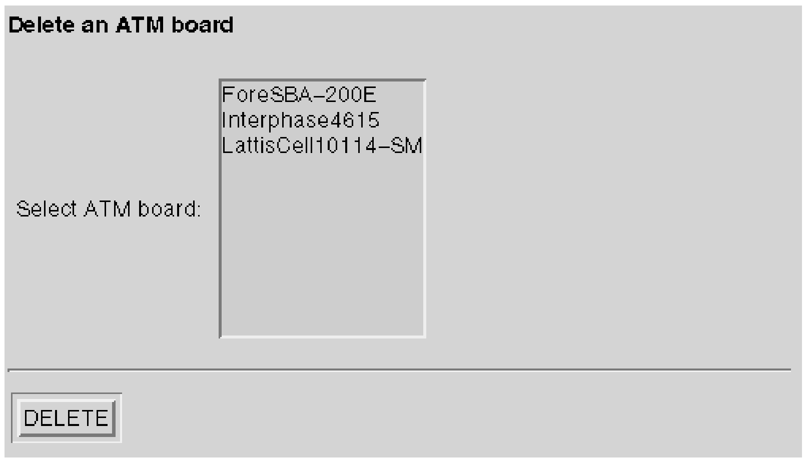


Figure 5.8: Selection-list generated by `w3-msql`

Although `w3-msql` provides a convenient way to implement dynamic forms, it's current version 1.0 provides only simple output functions. For complex TOMCAD functions that require control structures and inter-process communication as well as database access, another MiniSQL interface had to be used. MiniSQL is a very popular database product. Many additional interfaces have been developed (like Perl-, Tcl-, or Java-interfaces). Version 2 of `w3-msql` will include enhanced features (e.g control structures) that will allow to shift more functionality from the CGI to HTML.

Conclusion and future work

The design and implementation of TOMCAD showed that web-based software can hide the heterogeneity of a network. A number of enhanced web-features combined, allowed to implement a simple, uniform and comfortable user-interface. This interface proved to be suitable for basic connection management tasks. The page-oriented output and the lack of control during the execution, however, required to implement simple transaction-like operations. Keeping the management functionality at a central point in the network allowed to configure different devices in a uniform manner. However, this concept turned out to be awkward for continuous traffic monitoring. The employment of applets was a solution that provided client-side functionality for direct device access integrated in the web. Applets proved to be a promising concept to integrate additional client-server applications into the web.

TOMCAD helps to manage virtual connections in a PVC-based ATM network. It is meant to be a temporary solution for the time until ATM management standards are finalized and implemented.

List of Figures

Bibliography

[1] A. Alles. ATM Internetworking. Technical report, Cisco Systems, Inc., 1995.

[2] ATM Forum. *ATM User-Network Interface Specification 3.0*, 1993.

[3] ATM Forum. *LAN Emulation over ATM Specification - Version 1*, February 1995.

[4] P.P.S. Chen. The entity-relationship model - toward a unified view of data. *ACM Transactions on Database Systems*, March 1976.

[5] J. Heinanen. *Multiprotocol Encapsulation over ATM Adaptaion Layer 5.* Internet Engineering Task Force. RFC 1483, July 1993.

[6] I.S.Graham. *HTML sourcebook.* John Wiley & Sons, Inc, 1995.

[7] ITU-T (Recommendation M.3010). *Priciples for a telecommunications management network*, 1993.

[8] E. Johnston. BAGNet: A High Speed, Metropolitan Area, IP over ATM Network Testbed. Technical report, Lawrence Berkeley Laboratory, Univerity of California, Berkeley, January 1995.

[9] A.K. Larsen. The Next Web Wave: Network Management. *Data Communications*, January 1996.

[10] A.K. Larsen. Weaving the Management Web. *Data Communications*, January 1996.

[11] M. Laubach. *Classical IP and ARP over ATM.* Internet Engineering Task Force. RFC 1577, January 1994.

[12] M.T. Rose. *The Simple Book*. Prentice-Hall, 1991.

[13] NLS, New Line Software, Inc. *A Simple Network Management Executice Paradigm*, 1995.

[14] K.Carpenter P.Alexander. ATM Net Management - A Status Report. *Data Communications*, September 1995.

[15] SUN Microsystems, Mountain View, CA 1994. *The Java Language Specification*, 1995.

[16] T.Ritchey. *Java!* New Rider Publishing, 1995.

[17] Y. Inoue W.J. Barr, T. Boyd. The TINA Initiative. *IEEE Communications Magazine pp 70-76*, March 1993.