

Web-Based Customer Management of VPNs

Raouf Boutaba,¹ Walfrey Ng,² and Alberto Leon-Garcia³

The use of Virtual Private Networks (VPNs) is one of the major trends in the integrated broadband communications environment. Currently, the control and management of VPN resources is mainly supported by the provider of the bearer telecommunication services, and VPN customers have no control over these resources. The increasing importance of the broadband communication infrastructure in corporate operations and transactions is stressing the requirement for a customizable configuration, operation and management of VPN services. First, this paper discusses the evolution of VPN environments towards customized VPN configurations and goal-driven management of these VPNs. VPN service characteristics and management requirements are analyzed. Then, the paper introduces the proposed customer management architecture satisfying these requirements. The architecture is based on the multi-level virtualization of network resources through the partitioning of resources in the provider's shared communication infrastructure and the dynamic allocation of these resources to customers. A Web-based distributed approach is used for implementing the proposed customer management of VPNs.

KEY WORDS: VPN; resource partitioning; customer management; Web-based management.

1. INTRODUCTION

A Virtual Private Network (VPN) aims at offering a flexible and cost-effective transport service to a multi-site customer. As opposed to leased-lines-based private networks, the VPN service allows for a more efficient and flexible use of the common public network infrastructure so that communication resources can be easily adjusted according to the changing traffic requirements of customers.

¹Department of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. E-mail: rboutaba@bbcr.uwaterloo.ca

²IBM Lab, Toronto Lab, IBM Canada Ltd., Markham, Ontario L3R 9Z7, Canada. E-mail: wng@ibm.com

³Department of ECE, University of Toronto, Toronto, Ontario M5S 3G4, Canada. E-mail: alg@nal.utoronto.ca

Although resource sharing allows for a cost-effective communication infrastructure, it poses the problem of ensuring the communications' privacy. Privacy here includes both the characteristics of communication services, such as the addressing scheme and the integrity of the data being communicated.

The VPN service is commonly offered by a service provider to a number of service subscribers referred to as the "VPN customers" in this paper. The "VPN Provider" can be a public network provider or a third party, which uses bearer network facilities offered by one or several network providers. In traditional VPN environments, the provider of the bearer communication services manages the VPN resources, while VPN customers have no control over the resources allocated to them. The increasing importance of the broadband communication infrastructure in corporate operations and transactions is stressing the requirement for customizable configuration, operation, and management of VPN services.

To allow for a flexible and efficient operation and management of VPN resources in a multi-domain environment and to respond to the customizable management requirement, the VNRM (Virtual Network Resource Management) architecture has been designed. The VNRM architecture is generic in that it applies, through customization, to both the VPN providers and VPN customers domains. It is based on the virtual network (VN) concept introduced as a generalization of the VPN for supporting multi-levels virtualization of network resources. The purpose of a VN is to allow a "*soft networking environment*". Soft networking environment means flexible resource capacity, configurable network topology, and programmable resource/network management functions.

The configuration of VPNs and their customized management is effectively achieved through logical partitioning of MIBs. The *MIBlet* concept is used to provide abstract and selective views of the physical network resources allocated to VN customers. The abstract view hides the details of the resource interface that are not relevant to the customer's VNRM. The selective view restricts the access of the customer's VNRM system to those parts of the network resource allocated to this customer. Both hard and soft partitioning of resources are supported by the *MIBlet* concept. Using *MIBlets*, a customer VNRM system, referred to as Customer Network Management Systems or CNRMS, enforces its own control policies; and a provider VNRM system implements additional arbitration and policing functions.

The implementation of the CNRMS is distributed to allow for efficient operation and management of the VPN. This is achieved by means of downloadable customer control and management software. The latter is hosted by a programmable controller at the level of each resource involved in providing the VPN. The programmable controller acts autonomously on behalf of the customer to monitor and control those parts of the resource allocated to the customer through the resource representation in the *MIBlet*. It manages the execution of

management software using a script MIB and interacts with the MIBlet controller through a standard SNMP interface. The human operator/manager in the customer domain accesses the CNRMS through a Java-enabled Web browser. SNMP, Java, and Web technologies are used for openness and portability of VNRMS systems and thereby customer management of VPN services.

This paper is structured as follows. Section 2 studies the characteristics of the VPN service, and analyzes the requirements for customer management of VPN services. Section 3 introduces the generic customizable CNRMS architecture and the partitioning of network resources using *MIBlets*. Section 4 describes the Web-based design of the distributed CNRMS architecture. Finally, Section 5 concludes the paper and points out future research directions.

2. REQUIREMENT FOR CUSTOMER MANAGEMENT OF VPNs

There are a myriad of definitions of Virtual Private Network (VPN) used in the networking community to describe a broad set of problems and solutions. Ferguson and Huston [1] define a VPN as a communication environment in which access is controlled to permit peer connections only within a defined community of interest. A VPN is constructed through some form of partitioning of a common underlying communication medium, where this underlying communication medium provides services to the network on a nonexclusive basis.

A VPN service is primarily useful for organizations (VPN customers) that wish to use public networks (MANs and WANs) to connect their various LANs for private purposes. Therefore, the VPN concept has to respond to the two conflicting requirements: (1) Allow for a cost-effective communication infrastructure through resource sharing. Compared to the approach of dedicated leased circuits, organizations reduce the cost of connecting geographically dispersed sites by establishing VPNs across a shared public network; (2) Allow for communication privacy. Although several organizations share a common communication infrastructure (public backbone network), they want their communication services to be within one closed environment isolated from all other environments.

Currently, customers of all ranges of large to medium and small enterprises are relying more than ever on VPNs to conduct their businesses. For that reason, they are either acquiring the appropriate management tools and qualified personnel to administrate and maintain their growing customer premises networks or outsourcing the management of their network resources to third parties. Moreover, customers are seeking to control and manage the VPN services they are subscribing to. There are several reasons for that. Above all is the possibility for customers to control and manage their VPNs according to their own policies reflecting their business goals. A VPN service provider cannot easily accommodate the various customers' large variety of service requirements. Customers may have different traffic requirements (data, voice, and video) with different prior-

ity schemes and performance characteristics. They often require different levels of security. Another important reason for customers to control and manage their VPN is to perform the necessary partitioning of the VPN resources among the different end-users and applications they support, and to implement their own policing mechanisms. Last but not least is the ability of customers to introduce new communication services if they have full control over the resources allocated to them in the provider's network nodes, and hence the possibility to introduce their own resource management algorithms. This trend has been recently strengthened by the emergence and wide acceptance of network programmability as the networking paradigm of the future.

A large effort is currently spent in both academia and industry to open the core network infrastructure and facilitate its programmability by providing the appropriate application programming interfaces. The following lists the ongoing works on related issues: the definition of open switching architectures [2]; the specification of open signaling protocols [3]; and the development of programmable and active networks [4]. This trend will bring new challenges to the control and management of network resources. One of the most critical problems that need to be addressed is the shared control and management of the network resources between several provider/customer domains, which may lead to conflicts. A few research groups are addressing these issues by means of resource virtualization in active and programmable networks [5, 6]. In general, the functions of each administrative domain (network providers/customers) and the interactions between the different domains have to be re-engineered.

3. VPN PROVISION AND CUSTOMIZED MANAGEMENT

A generic Virtual Network Resource Management (VNRM) Architecture was designed in the Network Architecture Lab at the University of Toronto for network virtualization [7]. The VNRM Architecture is based on the Virtual Network (VN) concept. A Virtual Network (VN) is defined as a collection of Virtual Network Resources (VNRs). A virtual network resource is defined as a logical subset of a physical network resource such as transmission bandwidth, buffer queues, switch processing power, address space, scheduling mechanisms and so forth. This resource virtualization can be generalized to create multiple VN layers through vertical spawning (see Fig. 1).

The generic VNRM architecture can be customized and applied to manage every VN layer and every sub-network in a network composition. However, for the sake of clarity the following discussion is restricted to a limited number of client/server relationship levels between VN providers and VN customers, and therefore "VN" and "VPN" are used interchangeably in the remaining of the paper. Typically, a number of Network Providers will play the role of Root-VPN providers. A VPN service provider will play the role of the Primary VPN

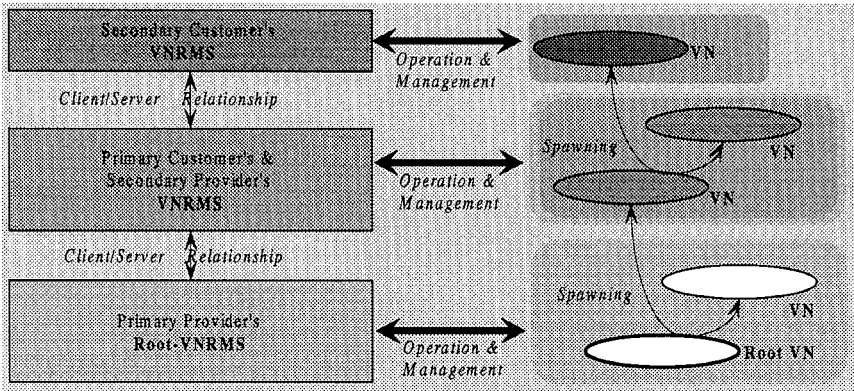


Fig. 1. Domain-based VNRMS systems.

customer (the secondary VPN Provider). The VPN service provider aggregates the network resources of multiple Network Providers and provides a single stop shopping to VPN customers. A VPN customer is a secondary customer and is typically a private company with a number of geographically dispersed branches. If uses the VPN provider services to interconnect its branches.

In this paper emphasis is put on the Web-based implementation of the VNRMS architecture, mainly on the customer perspective. This is referred to in the following as the Customer Network Resource Management System (CNRMS). Ultimately, VPNs are created for the use of customers who, in turn, provide network services to end-users. Depending upon network operation/management objectives, each customer may want to have full control over network operation algorithms and mechanisms for Fault, Configuration, Accounting, Performance and Security management, (FCAPS) routing, and signaling. This means that a CNRMS encompasses both traffic and network management aspects within a single framework. This is provided through programmable VNR Controller hosted by the Resource Agent at the level of the resource. With a Resource Agent, a VNR Controller controls a group of VNRs and interacts with the corresponding CNRMS for providing and managing customer domain network services. Figure 2 shows one Root-VN and two child VNs. Each and every VNR Controller is directly accessed by the corresponding customer control and management system (the CNRMS). Each VNR Controller serves its own CNRMS as an element of the virtual networking environment for the customer. During the lifetime of the group of VNRs, the VNR Controller performs usage control (or policing) to ensure conformance to the capacity contract in the use of the VNRs. By delivering abstract and aggregated information about status and connectivity of VNRs to the CNRMS, the Resource Agents

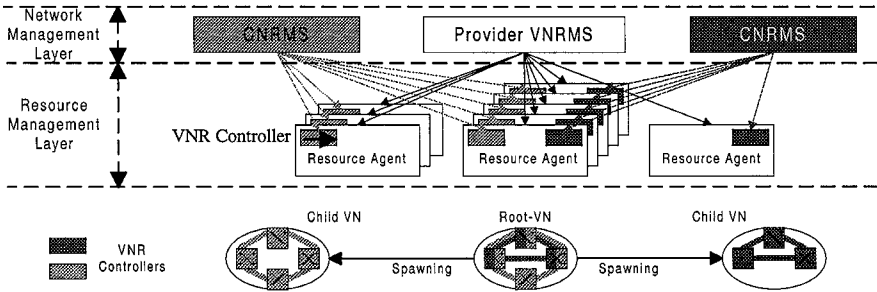


Fig. 2. Customer control of virtual networks.

hide the details of the resource information that are not relevant to the CNRMS. The operations of a VNR Controller voluntarily abide by rules and policies of its own Resource Agent, which is in the domain of the VPN service provider.

Programmability of Virtual Network Resources (VNRs) Controllers with customized algorithms and mechanisms is achieved through the dynamic binding of the VNRs to the customer's management system. The implementation of the dynamic system binding depends on the availability of management/control interfaces to resources (e.g., switches). The lack of open interfaces hampered the fast introduction of innovative network technologies and functions, and hence the network programmability. To stimulate faster development and support of networking applications, research and standardization efforts are currently dedicated to the specification of open switching/routing architectures, signaling protocols and binding interfaces. For the time being the functions of network resources are mostly implemented by rigidly built-in software, which accesses vendor-proprietary instrumentation to set up, monitor and control network connections. This has led to a redefinition of the role of network management through MIBs to incorporate features traditionally engineered into embedded software. For example ATM switches initially provided access to virtual channel instrumentation via SNMP MIBs and RSVP routers were controlled by management software. Currently, virtual LAN features are supported by switched LANs through management interfaces using MIBs to set up, monitor and control virtual LANs. The reason for that is the complexity of development of signaling protocols and associated software. More importantly, the time to market for management software is much less compared to that of signaling software.

We have chosen MIBs and SNMP as pragmatic tools to realize the VNRM architecture. Also, our goal is to provide IP over ATM VPN configurations using the network resources of our ATM testbed in which both the signaling software and the SNMP MIBs access and manipulate the same switch instrumentation. The MIBs provided with the ATM switches already incorporate a large number

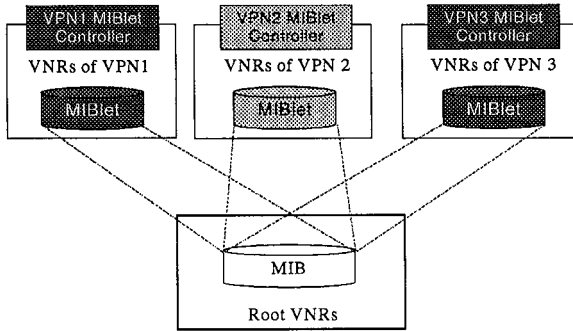


Fig. 3. The MIBlet concept.

of accessible configuration variables that can be used to effectively partition the ATM network to support multiple IP-based VPNs. The MIBlet concept allows such resource partitioning [8].

3.1. The MIBlet Concept

In order to allow a CNRMS to manage only a subset of resources in a network element, the MIB of the network element is logically partitioned into multiple MIBs we call *MIBlets*. The MIB of the network element can be identified with Root-VNRs, and the MIBlet can be identified with VNRs of a VPN. The Resource Agent is responsible for providing *abstract* and *selective* views to each CNRMS accessing it. An abstract view hides the details of the resource interface that are not relevant to the CNRMS. A selective view restricts the CNRMS to only access the resources allocated to it. To provide these views, instead of allowing a CNRMS to visualize all the managed objects in the MIB, the Resource Agent provides the CNRMS with only the managed objects that are relevant to the CNRMS. Also, the Resource Agent needs to control the access of the CNRMSs requests to manipulate the managed objects. This form of access control prevents the CNRMS from manipulating the managed objects in a way that violates its reservation condition. Figure 3 shows one Root-VN and two child VNs.

MIBlets and thereby VPNs result from the partitioning of the common underlying network resources (their MIB representations). Customer's VPN operation and management functions are implemented through *MIBlet controllers* located at every network node involved in the customer's VPN. MIBlet controllers implement the functionality of VNR controllers depicted in Fig. 2. In particular, they enforce the customers' access control and resource usage policing strategies and are invoked to set up, monitor and control the customer's

connections. Each and every MIBlet Controller is directly accessed by the corresponding CNRMS and serves as a delegated component for this CNRMS.

Different levels of granularities are used for the partitioning of the ATM switch resources to provide multiple MIBlets at the level of each ATM switch in our ATM testbed. These are: port, VPI/VCI space and bandwidth. Orthogonal to these partitioning levels are the types of partitioning. These may be *hard* partitioning leading to resource dedication or *soft* partitioning favoring resource sharing. A *hard VNR* is created with a specified amount of capacity allocated. A *soft VNR* is created with no explicit amount of capacity allocated. Further amount of resource capacity is allocated dynamically on demand up to a maximum amount. The capacity of a hard VNR may also be changed through management requests from the customer, which normally takes place in a longer time scale. The concept of a soft VNR plays an important role for demand-based dynamic capacity management of VPNs. It leads, however, to competition among VPNs, and thus necessitates appropriate mechanisms for admission control and resource usage policing. In the VNRM architecture these mechanisms are provided in both the provider and the customer management domains: the CNRMS enforces its own control policies; and a provider VNRMS implements additional arbitration functions.

4. WEB-BASED CUSTOMER MANAGEMENT SYSTEM IMPLEMENTATION

Installation of new versions of monitoring and control software in a network is complex and costly. Transparent and rapid downloading via Java-enabled Web browser constitutes an attractive alternative. Here, we present a Java-based implementation to demonstrate how the CNRMS architecture applies. Our implementation integrates a Web-based management interface, and is based on a Web-based distributed management system developed for monitoring large-scale networks [9]. In this implementation, programmable controllers are delegated by a CNRMS to monitor and control VPN resources. The monitoring and controlling activities are performed through MIBlet controllers.

The implemented prototype consists of three main tiers: a client which is basically a *Web Browser* providing the management interface; a *Management Server* which is the central component of the system and which implements the management applications; and finally a set of *Resource Agents*. The communications among these components are standard Internet protocols, which are HTTP, TCP and SNMP. For these communications, our prototype implementation uses a separate IP VPN spawned from our physical ATM network testbed. The structure of the prototype implementation is shown in Fig. 4.

In a CNRMS, human operators launch management operations through Web browsers. The management server contains a web server which continuously

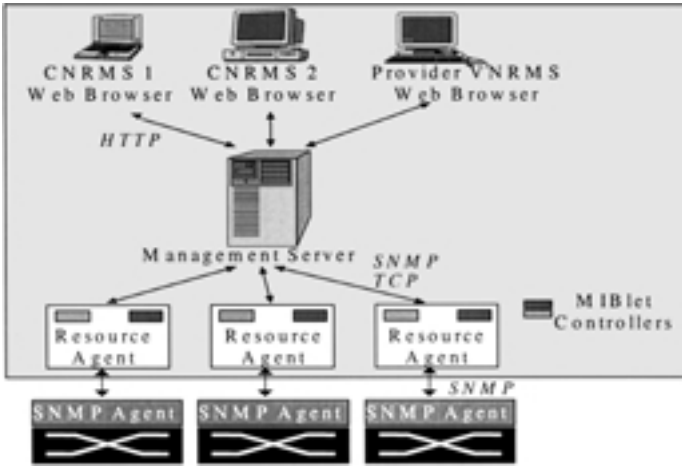


Fig. 4. Structure of the prototype implementation.

listens on a given communication port to receive HTTP requests from browsers. The management server carries out management operations on the customer's VPN resources according to the HTTP requests by sending SNMP requests to the appropriate MIBlet controllers. In response to HTTP requests the Management server may also download management software packages to the resource agents to be executed on the managed resources. A standard SNMP agent represents each managed resource. All components of the system are implemented in Java, mainly for the portability feature of the language.

4.1. Client (Web Browser)

The client is a standard Web browser where the interfaces will be displayed. These interfaces change during the evolution of the management function by downloading HTML pages and applets from the web server. To access the management function, the client needs to know the URL address of the workstation hosting the web server and possibly the communication port number of the server. Once the connection is established, the client launches a discovery mechanism, which displays in an HTML page the Resource Agents managed by the system. The client can then carry out different management functions on the Resource Agents. The requests issued by the client to the web server are HTTP requests.

4.2. Management Server

Currently, the Management Server can carry out functions such as creating a MIBlet, terminating a MIBlet, delegating management tasks, and download-

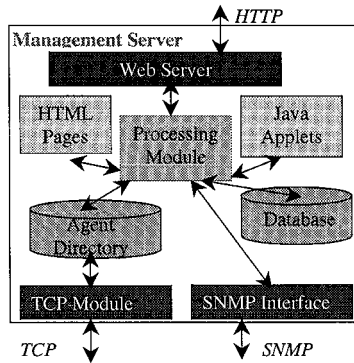


Fig. 5. Management server architecture.

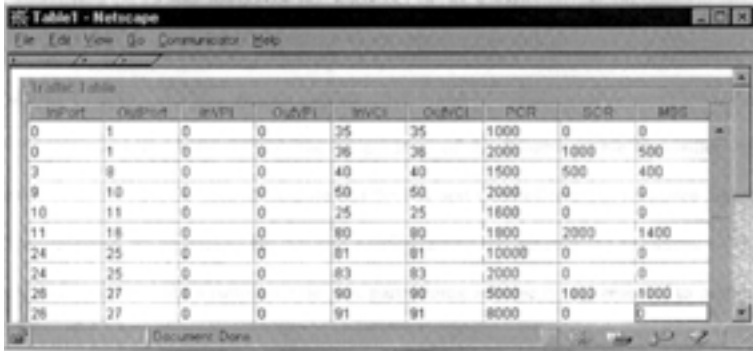
ing signaling/routing controller software. Figure 5 shows the architecture of the Management Server and the server's modules are described.

4.2.1. Web Server

The Web Server module listens continually on its communication port of the HTTP requests issued by the clients. It deals with the client requests according to two schemes. In the first scheme, the Web server creates HTML pages dynamically by integrating applets or using the existing HTML pages in response to client requests. In the second scheme, the Web server invokes the processing module to perform the necessary management operations.

4.2.2. Processing Module

If the Web server receives a request that requires a particular processing, the Web server will then invoke the processing module. The main responsibility of the processing module is to interpret client requests and translate them into SNMP commands to be forwarded to the Resource Agents through the SNMP interface module. Management information related to the Resource Agents is stored in a database and is used by the processing module to dynamically create HTML pages which are then forwarded to the Web server. For example, the Web server may request the processing module to retrieve the information of the connections established by a MIBlet. After the information is retrieved and stored in the database, the processing module uses this information to dynamically build an HTML page, which displays in a tabular form the input port, the input VPI/VCI, the output port, the output VPI/VCI, and the bandwidth allocation for each connection established by the MIBlet. Figure 6 shows an example of such an HTML page which also gives the traffic information for each of these connections in terms of the Peak Cell Rate (PCR), the Sustainable Cell Rate (SCR), and the Maximum Burst Size (MBS).



InPort	OutPort	InVPI	OutVPI	InVCI	OutVCI	PCR	SCR	MSS
0	1	0	0	35	35	1000	0	0
0	1	0	0	36	36	2000	1000	500
3	8	0	0	40	40	1500	500	400
9	10	0	0	50	50	2000	0	0
10	11	0	0	25	25	1800	0	0
11	18	0	0	80	80	1800	2000	1400
24	25	0	0	81	81	10000	0	0
24	25	0	0	83	83	2000	0	0
28	27	0	0	90	90	5000	1000	1000
28	27	0	0	91	91	8000	0	0

Fig. 6. Example generated HTML page containing the traffic table for a MIBlet.

4.2.3. SNMP Interface

The SNMP interface is invoked by the processing module to handle SNMP messages. It is the interface between the processing module and the Resource Agents. The messages used by this interface are standard SNMP messages. The information received from the Resource Agents is stored in the database and/or sent directly to the processing module. To trigger the execution of a script or a signaling/routing controller, the SNMP interface sends a set-request to the Programmable Controller in the Resource Agent. To request for MIBlet creation, re-configuration or termination, the SNMP interface sends a set-request to the Request Controller in the Resource Agent. The format for these set-requests is described in the subsequent sections.

4.2.4. TCP Module

The first task of the TCP module is to transfer management scripts or controller software to the appropriate Resource Agents. Another task is to record the information related to registered Resource Agents. Once a Resource Agent becomes active, it informs the Management Server and forwards some relevant information such as its IP address and communication port number.

4.3. Resource Agent

There are four functional building blocks in the Resource Agent: (1) Request Controller, (2) MIBlet Controller, (3) Resource Controller, and (4) Programmable Controller. The first three building blocks focus on how a MIBlet provides CNRMS with a limited view of the switch's resources; while the Programmable Controller is responsible for the customization and programmability of the VPN. In the Resource Agent, MIBlet Controller and Programmable Controller are instantiated for each VPN. Figure 7 shows the design of the Resource Agent.

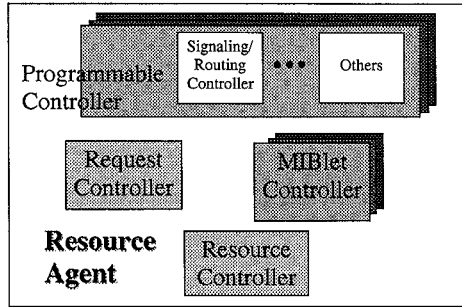


Fig. 7. Architecture of Resource Agent.

4.3.1. Programmable Controller

The Programmable Controller is responsible for receiving a variety of functional software packages downloaded from the CNRMS, and these software packages are executed by the Programmable Controller. The Programmable Controller is a key building block that allows customers to have full customization of network control, and provides the Resource Agent with the intelligence for an autonomous operation. Each CNRMS has its own Programmable Controller in the Resource Agent. The Programmable Controller can download software packages with different functionality. An example is the signaling and routing control software required for each Virtual Network Switch, which is a logical subset of a switch. Other possible software packages can be used for congestion control [10], video stream management [11], and FCAPS. The Programmable Controller contains four modules: (1) SNMP command responder, (2) Scripts MIB, (3) Load module, and (4) Execution module. The modules involved in the Programmable Controller are illustrated in Fig. 8.

SNMP Command Responder: The SNMP Command Responder receives SNMP requests from the Management Server. SNMP get-requests lead to the retrieval of one or more values from the MIB/MIBlet variable identified by the OID. The SNMP set-request is used to: (1) update the value of the indicated

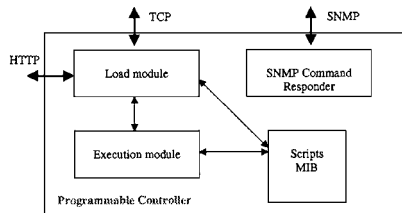


Fig. 8. Modules involved in Programmable Controller.

variable: (2) download management scripts from a remote location; (3) trigger the execution of a script; and (4) open a TCP connection to receive a script from the Management Server.

Scripts MIB: The Scripts MIB [12] contains references to executable objects, which allow when invoked to receive, download or execute scripts. These object references are defined as: *ExecuteClass* (responsible for loading and executing a script); *ReceiveClass* (responsible for opening a TCP connection and receiving scripts from the TCP module in the Management Server); *DownloadClass* (responsible for downloading scripts from remote locations). Each of these object references has an OID in the MIB which is known by the Management Server. When receiving a set-request with one of these OIDs, the corresponding object reference is activated and the appropriate operation is executed.

Execution Module: When the SNMP Command Responder receives a set-request pointing to the OID that corresponds to the “ExecuteClass” object reference, it invokes the execution module. The name of the script to be executed is specified in the SNMP set-request as a parameter. The execution module first checks if the script exists in the Programmable Controller. If it does not exist, the Programmable Controller sends a response to notify the Management Server that the script is not available. Otherwise, the execution module loads and executes the script using the Java Class Loader.

Load Module: The Load module is responsible for transferring scripts using one of two schemes: direct transfer mode, or indirect transfer mode. In the direct transfer mode, when the Management Server wants to transfer a script to the Programmable Controller, it forwards an SNMP set-request to the target Programmable Controller. The set-request encapsulates the OID corresponding to the “ReceiveClass” object reference. This object in turn invokes the load module for the script transfer. The load module then opens a TCP connection and sends a READY message to the Management Server to initiate the transfer. When the script is correctly received, the load module closes the TCP connection. In the indirect transfer mode, after receiving the set-request encapsulating the OID corresponding to the “DownloadClass” object reference, the Programmable Controller invokes the load module for indirect transfer. The load module then downloads the script from the remote site using the URL specified in the set-request.

4.3.2. Request Controller

The Request Controller is invoked whenever a MIBlet creation request generated by a CNRMS is received. The Request Controller examines the request against predefined policies. An example policy is to limit the maximum amount of bandwidth that can be reserved by any CNRMS. If the request does not comply with the policies, the Request Controller sends a response to the CNRMS indicating that the request is rejected. Otherwise, the Request Controller checks to see if there are enough resources to create the requested MIBlet. If there

are not enough resources, the controller sends a reject message to the CNRMS. Otherwise: (a) a service address, which is to be used by the CNRMS to communicate with its MIBlet Controller, is assigned to the newly created MIBlet Controller; (b) a MIBlet creation response, encapsulating the service address of the MIBlet Controller, is sent to the CNRMS indicating that the request is accepted; and (c) reservation parameters are fed into the newly created MIBlet Controller.

The Request Controller contains three modules: (1) SNMP Command Responder, (2) Scripts MIB, and (3) Registration module. The SNMP Command Responder is responsible for receiving MIBlet creation, re-configuration, and termination requests from the Management Server. These requests are forwarded in the form of SNMP set-requests. The object references in the Scripts MIB are defined as follows: *MIBletCreateClass* (for initiating a new MIBlet Controller); *MIBletReconfClass* (for reconfiguring a MIBlet Controller); *MIBletTerminateClass* (for terminating a MIBlet Controller).

To create a MIBlet Controller, the Management Server sends a request pointing to the OID that corresponds to the “MIBletCreateClass” object reference. The reservation parameters of the MIBlet are specified in the request. Upon receiving a MIBlet creation request, the “MIBletCreateClass” object initiates the registration module, which then examines if there are enough resources to create the requested MIBlet. If there are enough resources, a response encapsulating the IP address and the communication port number of the newly created MIBlet is sent to the Management Server to indicate that the request is accepted. Also, the reservation parameters are recorded in a database. The MIBletCreateClass parameters are: (1) port number; (2), the number of VPIs on the specified port; (3) the number of VCIs on the VPIs of the specified port; (4) the amount of hard and soft bandwidth.

A CNRMS may require a different amount of bandwidth and network topology at different times, based on the timely profile of its network traffic. The MIBlet re-configuration request allows the CNRMS to change the configuration during the lifetime of the MIBlet. To re-configure a MIBlet Controller, the Management Server sends a request pointing to the OID that corresponds to the “MIBletReconfClass” object reference. The registration module is then invoked to examine if there are enough resources for the re-configuration. If there are enough resources, the reservation parameters of the MIBlet in the database are replaced with the re-configuration parameters, and a response is sent to the Management Server to indicate that the request is accepted. The parameters that need to be specified in the message are the same as those specified in the MIBlet creation request.

To terminate a MIBlet Controller, the Management Server sends a request that points to the OID corresponding to the “MIBletTerminateClass” object reference. This object then initiates the registration module to clear the reservation

parameters of the MIBlet in the database. The Service Address of the MIBlet that the CNRMS wants to terminate is the parameter specified in the request.

4.3.3. MIBlet Controller

Once the MIBlet creation request is accepted by the Request Controller, the CNRMS can send requests, such as Virtual Channel segment establishment and inquiry for switch information, to its MIBlet Controller. The latter maintains information about ports and the ranges of the Virtual Channel and Virtual Path identifiers (VPI/VCI) that are reserved by its CNRMS. It also maintains the amount of hard and soft bandwidth reserved in each port. For requests not related to bandwidth, the controller checks to see if the requests comply with the reservation condition. For bandwidth related requests, the controller checks whether or not the reserved hard bandwidth is already allocated. If hard bandwidth is available, the request is then sent to the switch. If all the hard bandwidth is allocated while soft bandwidth is still available, the MIBlet Controller will send the request to the Resource Controller in order to compete for shared bandwidth. If both hard and soft bandwidth are allocated, the request is rejected and a negative response is sent to the CNRMS.

The MIBlet definition provided by the MIBlet Controller is very similar to the switch's MIB definition provided by the switch vendor. The MIBlet definition keeps all the important MIB groups in the switch's MIB definition and omits the MIB groups that are not relevant to the CNRMS. Moreover, additional MIB groups can be added to the MIBlet definition. An example of an added MIB group, which contains the information about the reserved resources is the portResTable. Each row in this table contains the max VPI, the min VPI, the max VCI, the min VCI, the hard bandwidth and the soft bandwidth in a reserved port. The index used to pick a specific row out of the portResTable is the port number. The managed objects in this group are not physically implemented in the switch's actual MIB. When the Management Server sends a get-request to get the values from this MIB group, the MIBlet Controller will immediately provide the Management Server with the get-response and does not need to pass the get-request to the actual MIB in the switch.

4.3.4. Resource Controller

The main responsibility of the Resource Controller is to act as an arbiter when MIBlet Controllers compete with each other for shared resources. When the MIBlet Controllers need to compete for shared resources on behalf of their CNRMSs, the MIBlet Controllers will send the requests to the Resource Controller, Section 4.3.3, which presents the functions of the MIBlet Controller, describes when the MIBlet Controller will send the request to the Resource Controller.

The Resource Controller maintains the shared bandwidth for each port. Every time the controller receives a request for connection establishment, it com-

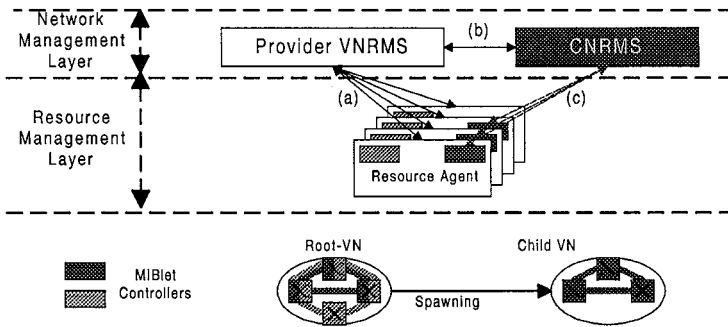


Fig. 9. Interfaces for provision and customization.

putes the aggregate bandwidth usage for supporting this new connection. When the aggregate bandwidth usage reaches the amount of shared bandwidth, this implies that all the shared bandwidth is allocated and further requests will be rejected. If the request is accepted, (a) the request is sent to the switch; (b) the aggregate bandwidth is computed for deciding whether or not to accept this new connection. If the connection request is rejected, the Resource Controller will notify the CNRMS.

4.4. Communication Interfaces

In order to provide, customize and manage a virtual network, the CNRMS, the VPN provider management system (referred to as the provider's VNRMS in the following) and the Resource Agents need to communicate with each other (see Fig. 9). Three categories of interfaces are specified: (a) between the provider's VNRMS and Resource Agents; (b) between the CNRMS and the provider's VNRMS; and (c) between the CNRMS and Resource Agents.

4.4.1. Communication Between Provider VNRMS and Resource Agents

There are six messages involved in the communication between the provider's VNRMS and Resource Agents: MIBlet creation request/response, MIBlet re-configuration request/response, and MIBlet termination request/response. The requests are sent from the provider's VNRMS to the Request Controller in the Resource Agent. For example, to spawn a VPN, a MIBlet is created at the level of each node involved in the VPN. In the case of an ATM switch, VPI/VCI space and bandwidth are partitioned and allocated to the MIBlet. After each switch's MIBlet is created, a VPN can simply be represented as a collection of the newly created MIBlets. The reservation information of the MIBlet creation request is described in Section 4.3.2 and is obtained by the management server from the CNRMS Web-based management interface as shown in Fig. 10. The

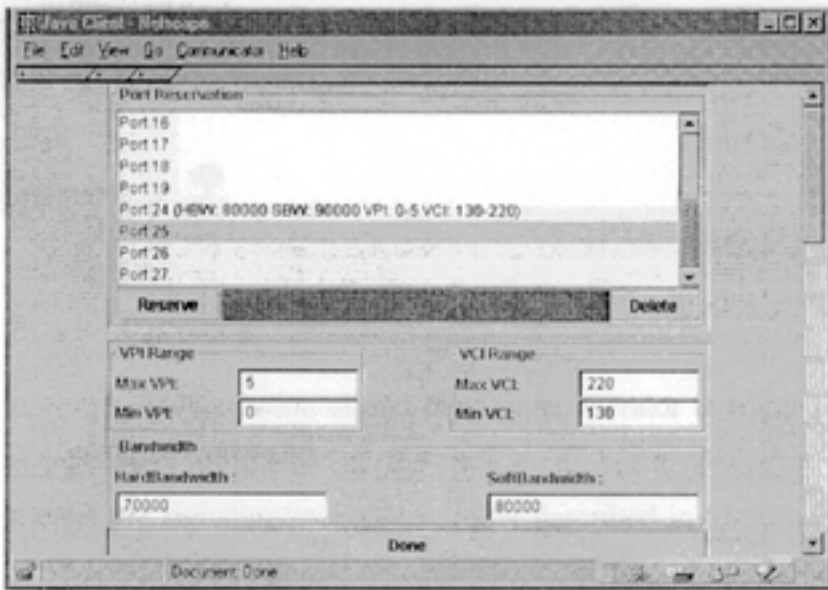


Fig. 10. Web-based interfaces for provision and customization.

MIBlet creation response specifies, if the request is accepted, a Service Address that is used by the CNRMS to communicate with its MIBlet Controller.

4.4.2. Communication Between CNRMS and Provider's VNRMS

The message in the communication between a CNRMS and a provider's VNRMS are related to creation, re-configuration and termination of the whole VPN. For example, a VPN creation request is sent by the CNRMS to the provider's VNRMS. It specifies the location of the customer's systems (switches/routers) that plan to connect to the requested VPN and the bandwidth requirements for the connections among the customer's systems. The VPN creation response specifies, if the request is accepted, the service addresses of all the MIBlet Controllers involved in the VPN. A VPN reconfiguration request message is also used at this interface. It has the same parameters as those of a VPN creation request. Finally, a VPN termination request is used by the CNRMS to terminate its VPN. The provider's VNRMS will then send MIBlet termination requests to all the Resource Agents involved in the VPN to terminate the MIBlets.

4.4.3. Communication Between CNRMS and Resource Agents

To operate and manage the VPN, the CNRMS interacts with the MIBlet Controllers in the Resource Agents. To manage the MIBlet and monitor traffic,

a management protocol, such as SNMP or CMIP, is used. In our implementation SNMP is used. Moreover, to customize the VPN and allow autonomous operations of the Resource Agents, CNRMS downloads appropriate software packages to the associated Programmable Controllers in the Resource Agents using the TCP interface.

The CNRMS accesses the MIBlet as if it was directly accessing the MIB in the switch. However, the CNRMS can only access the values related to the reserved resources. The MIBlet Controller will provide the CNRMS with a MIBlet definition that defines the structure of the MIBlet objects. The MIBlet definition keeps all the important MIB groups in the vendors' switch MIB definition such as: the *Port Group*, the *Path Group*, and the *Channel Group*. Other MIB groups that are not relevant to the CNRMS are omitted in the MIBlet definition. Moreover, extra MIB groups can be added to the MIBlet definition.

4.5. Experimental Testbed

In order to configure the VPN for different control architectures, different signaling/routing controllers are required. Our testbed spawns three virtual networks from the physical ATM network. As illustrated in Fig. 11, a first VPN serves as an Integrated Services IP (IS-IP) network, a second VPN serves as an ATM network, and a third one serves as a Best Effort IP (BE-IP) network. ISAC (Integrated Services Internet with RSVP over ATM short-Cuts) [13] is used as the signaling/routing controller for the IS-IP virtual network. It is responsible for routing and RSVP signaling. It also performs parameter mapping which maps the QoS parameters from IS-IP model to appropriate ATM services. The signaling/routing controller used for the ATM VPN is the ATM signaling/routing controller [14] developed according to the ATM signaling standard. The BE-IP VPN is required for the operation of the ISAC controller because ISAC switch controllers employ RSVP as a signaling protocol and RSVP delivers control messages through BE-IP services. For the time being, BE-IP control mechanisms and services provided by our ATM switch are used for the control of the BE-IP VPN. In other words, we are exercising programmability and resource allocation flexibility for IS-IP VPN and ATM VPN only.

In this ATM testbed, partitioning at the ATM port, Virtual Path and Virtual Circuit levels as well as bandwidth (hard and soft), are supported to provide VPN services with differing QoS requirements and control schemes. For a detailed description of the defined partitioning schemes, the reader is referred to [8]. For a description of the implementation of the various IP-based VPN provided over the ATM testbed, namely the IP Switching based VPN using ISAC and classical IP over ATM based VPNs, the reader is referred to [15].

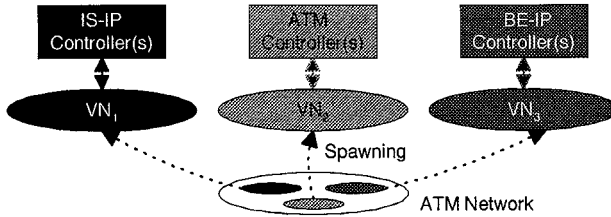


Fig. 11. Partitioning of the ATM Testbed.

5. CONCLUSIONS

The paper presented the CNRM system for Web-based Customer control and management of VPNs. The CNRMS is structured into a network management layer and a resource management layer. At the network management layer, network-wide customized operation and management functions are applied to the VPN resources. Resource Agents in the resource management layer handle the partitioning and allocation of resources to customers. They also host programmable controllers belonging to various CNRM systems, enforcing customers' policies for the control and management of the resources allocated to these customers. Using the MIBlet concept, we have shown how we can effectively partition a network into VPNs through simple, open and largely deployed MIB interfaces. MIBlets can be rapidly implemented without requiring new efforts for the definition and standardization of new dynamic binding interfaces. They can be deployed at a large scale in current networks and in future programmable networks that will be more likely composed of programmable and non-programmable nodes. Programmable controllers within the Resource Agent are introduced for downloading and executing the customer's control and management software. Executing delegated control and management functions at the same level as the MIBlets improves the response time for customer management applications.

In the current implementation, the dynamic configurability of the resources is enabled through the downloading of management software to autonomous Resource Agents. There is no architectural change required for existing control software to be deployed. Rather, the Resource Agent transparently fulfills partitioning of resources, arbitration of resource usage, and access control. In this way, new control and management environment. However, the physical separation of the control software from the switching hardware introduces a few extra communication procedures, which may impact the performance of the overall system. To achieve a programmable networking and management environment with less performance penalty, our future implementation will integrate the control and management functions within the active switch being developed in the Lab using programmable hardware [16].

REFERENCES

1. Paul Ferguson and Geoff Huston, What is a VPN?, Whitepaper cisco systems and Telstra Internet, Revision 1, April 1998. <http://www.employees.org:80/~ferguson/>, under "Presentations, Slideware, and Assorted Cruft".
2. *Proceedings of the Second IEEE Conference on Open Architectures and Network Programming (OPENARCH'99)*, New York, March 1999.
3. *Proceedings of the Workshop on Open Signaling for ATM, Internet and Mobile Networks (OPENSIG'98)*, Toronto, Canada, October 1998.
4. D. Tennenhouse, J. Smith, W. Sincoskie, D. Wetherall, and G. Minden, A survey of active network research, *IEEE Communications Magazine*, Vol. 35, No. 1, pp. 80–86, January 1997.
5. A. T. Campbell, M. E. Kounavis, D. A. Villela, J. Vicente (Intel), K. Miki (Hitachi), H. G. De Meer, and K. S. Kalaichelvan (Nortel), Spawning Networks, *IEEE Network Magazine*, pp. 16–30, July/August 1999.
6. Marcus Brunner and Rolf Stadler, Service management in multiparty active networks, *IEEE Communications Magazine*, March 2000.
7. A. Do-Sung Jun and A. Leon-Garcia, Virtual network resources management: A divide-and-conquer approach for the control of future networks, in *Proceedings of IEEE Global Telecommunications Conference (GlobeCom'98)*, Sydney, Australia, pp. 1065–1070, November 1998.
8. W. Ng, A. Do-Sung Jun, H. Chow, R. Boutaba, and A. Leon-Garcia, MIBlets: A practical approach to virtual network management, in *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM'99)*, Boston, Massachusetts, pp. 201–216, May 1999.
9. A. Ghalamallah and R. Boutaba, Implementing a distributed Web-based management system in Java, *IEEE ITS/SBT'98*, São Paulo, Brazil, August 7–10 1998.
10. A. Mehaoua, Y. Iraqi, and A. Ghalamallah, A self-regulating congestion control scheme using an intelligent multi-agent architecture, *Proc. IFIP/IEEE MMNS'97*, Montreal, Canada, July 1997.
11. E. Amir, S. McCanne, and H. Zhang, An application-level video gateway, *Proceedings of ACM Multimedia '95*, November 1995.
12. D. Levi and J. Schoenwaelder, Definitions of managed objects for the delegation of management scripts" RFC 2592, May 1999.
13. HungKei Keith Chow and Alberto Leon-Garcia, Implementation and performance evaluation of ISAC: Integrated service internet with RSVP over ATM shortcuts, *ICC'98*, June 1998.
14. Teshu Flower, Implementation of a signaling/routing controller for dynamic resource allocation over an ATM network, B.A.Sc. Thesis, Faculty of Applied Science and Engineering, University of Toronto, April 1999.
15. W. Ng., R. Boutaba, and A. Leon-Garcia, Provision and customization of ATM virtual networks for supporting IP Services, *Proceedings of IEEE ATM Workshop*, Kochi, Japan, May 1999.
16. M. R. Hashemi and A. Leon-Garcia, A RAM-based generic packet switch with scheduling capability, *IEEE BSS'97*, Taipei, December 1997.

Raouf Boutaba is a Professor in the Department of Computer Science of the University of Waterloo. Before that he was a Professor in the Electrical and Computer Engineering of the University of Toronto. From 1995 until late 1997 he built and was the Director of the Telecommunications and Distributed Systems Division in the Computer Science Research Institute of Montreal. He has been an adjunct Professor at the University of Montreal since 1995. Between 1990 and 1995 he was technical leader of EC-funded ESPRIT and ACTS projects. He conducts research in integrated network and systems management, wired and wireless multimedia networks, and quality of service

control in Internet networks. He founded and was the general chair of the IFIP/IEEE International Conference on the Management of Multimedia Networks and Services in 1997.

Walfrey Ng received the B.A.Sc. and M.A.Sc. in Electrical and Computer Engineering from the University of Toronto in 1997 and 1999, respectively. His research focuses on the Design and Management of ATM Virtual Network for IP Services. Since 1999, he has been a system engineer at IBM Lab (IBM Canada Ltd.).

Alberto Leon-Garcia is a Professor in the Department of Electrical and Computer Engineering of the University of Toronto and he currently holds the Nortel Institute Chair in Network Architecture and Services. He is a fellow of the IEEE “For contributions to multiplexing and switching of integrated services traffic”. He teaches undergraduate and graduate courses in communication networks, and conducts research in resources management of broadband networks and service end systems, switch and router design, Internet performance, and wireless packet access networks. He is the author of the textbooks *Probability and Random Processes for Electrical Engineering* (Addison-Wesley), and co-author of *Communication Networks: Fundamental Concepts and Key Architectures* (McGraw-Hill).