ETH zürich

ETH Library

The impact of active networking technology on service management in a Telecom environment

Report

Author(s):
Brunner, Marcus; Stadler, Rolf

Publication date:
1998-08

Permanent link:
https://doi.org/10.3929/ethz-a-004287915

Rights / license:
In Copyright - Non-Commercial Use Permitted

Originally published in:
TIK Report 55

This page was generated automatically upon download from the ETH Zurich Research Collection.
For more information, please consult the Terms of use.

The Impact of Active Networking Technology on Service Management in a Telecom Environment

TIK-Report, No. 55, August 1998

Marcus Brunner[*] and Rolf Stadler[**]

[*]Computer Engineering and Networks Laboratory, TIK
Swiss Federal Institute of Technology Zurich (ETH)
Gloriastr. 35, CH-8092 Zurich, Switzerland
E-Mail: brunner@tik.ee.ethz.ch

[**]Center for Telecommunications Research (CTR)
and Department of Electrical Engineering
Columbia University
New York, NY 10027-6699
E-Mail: stadler@ctr.columbia.edu

Abstract

Active networking, where network nodes perform customized processing of packets, is a rapidly expanding field of research. This paper is based on the assumption that active networking technology will mature to a point where it can be commercially deployed on a larger scale. We investigate the realization of service provisioning and service management in a telecom environment that is based on active networking technology, primarily with respect to customer-provider interactions. Compared to conventional networking technology, active networking concepts enable additional flexibility in supporting management tasks. We outline a framework that allows customers, on the one hand, to access and manage a service in a provider's domain, and, on the other hand, to outsource a service and its management to a service provider. Our framework has the properties of supporting (1) generic, i.e., service-independent, interfaces for service provisioning and management, and (2) customized service abstractions and control functions, according to a customer's requirements. Further, we describe how some of the key concepts of this framework can be realized in an active networking testbed that we are in the process of building.

Keywords

Management of Active Networks, Service Provisioning, Service Management, Customer Control, Outsourcing, Network Architecture

1 Introduction

In this paper, we investigate the problem of service provisioning and management in a telecom environment that is built using active networking technologies. Based on the fact that an active network node is able to run different execution environments concurrently, and that it can be accessed via a generic service interface, we explore the options for realizing the task of service provisioning and management, primarily with respect to customer-provider interactions.

In today's telecom environments, customers and service providers typically interact via two interfaces--the management interface and the service interface. The management interface, based on standardized management protocols, is generally used for service provisioning and service management. The customer accesses the service through the service interface. Both interfaces are service specific; a provider has to support different service and management interfaces for each service offered, which makes the introduction of new services dependent on standardization, and therefore time consuming and expensive.

The introduction of active networking technology in telecom environments, characterized by the use of active networking nodes as network elements, will change the role of these interfaces in two ways. First, using active networking nodes enables the definition of a generic service interface for network services, based on the concept of *active packets* (see Section 3). In addition, this service interface can be used by the customer for service management interactions, i.e., for operations related to the installation, supervision, upgrading and removal of a specific service. Therefore, service management operations can be performed by a customer without interaction with the provider's management system. Second, the management interface, i.e., the interface through which the customer and the provider management systems cooperate, can be restricted to the task of service provisioning. Similar to the service interface, the management interface can be kept generic; it relates to a generic service abstraction that allows for installing and running a large class of network services.

In a telecom environment, customers and providers interact in two different ways. The first way can be described by a set of functions offered by the provider to the customer through a specific service abstraction, such as a virtual network. The second way is known as outsourcing, whereby a provider configures and runs a service to be utilized by the customer. In today's environments, the first type of interaction is generally constrained to a simple, high-level service abstraction offered by the provider, which gives the customer limited management capabilities, mostly constrained to monitoring. Many customers favor this approach, since it keeps the complexity of their control and management systems low. However, there are customer groups with specific requirements, such as military organizations or large companies, which prefer more detailed service abstractions, combined with control and management capabilities inside the provider's domain. To meet the demands of these customers, new service abstractions are needed, such as the one in [1]. The second type of interaction, the case of outsourcing, generally gives a provider full access to network elements in the customer domain, which can be seen as a fine-grained model, associated with powerful management capabilities.

The introduction of active networking technology will change both types of interactions described above. The granularity of a service abstraction and its related control capabilities can be chosen, ranging from a very limited, constrained service model to a very detailed one. As we will show in the paper, a generic service abstraction, the *Virtual Active Network*, can be defined. A service instance based on this abstraction can be configured at the time of service provisioning, and it can then be refined according to a customer's specific requirements.

The paper is organized as follows. Section 2 surveys current efforts on active technolo-

gies for network management. Section 3 introduces the concept of active networking. Section 4 outlines our framework for service management and provisioning in a telecom environment that is based on active networking technologies. Some Scenarios illustrating our framework are presented in Section 5. Section 6 describes an active networking platform we have built for experimenting with active networking concepts, and it contains the design of an active networking node with respect to supporting management operations. Section 7 summarizes the contributions of this paper and gives an outlook on further work.

2 Related Work

Network management research related to active technologies has only recently been started. Many approaches taken today can be seen as a generalization of the concept of *management by delegation* [2]. They are focusing on using mobile code for building an active management middleware, i.e., a software layer between the management applications and the managed objects (MOs) that represent physical or other local resources. In [3] new classes of MOs, called active managed objects (AMOs), are proposed for event discrimination and aggregation of monitoring data. The behavior of AMOs is defined in a scripting language. The scripts encoding this behavior stored as values of AMO attributes. Therefore, the behavior of AMOs can be changed using standard management protocols. A similar approach is pursued in [4], where the concept of a dynamic MO is proposed, in order make Q-adapters programmable. In contrast to these approaches, this paper assumes an *underlying* active networking infrastructure, and its contributions are based on exploiting the capabilities of active networking nodes.

One part of the Netscript project [5][6] deals with management of active networks. In that project, a platform for programming network services is being built. These services can be automatically instrumented for management purposes, and corresponding MIBs can be generated. During operation, services can be managed through those MIBs. Contrary to the Netscript project, our work leaves open the question of service instrumentation in an active network environment, but it focuses on a flexible framework for supporting interactions between customers and providers.

Research approaches in the area of *programmable networks* focus on developing interfaces that facilitate flexible service creation and resource partitioning in a telecom environment. This work generally centers around a programmable control plane for broadband networks [7][8]. While the current research in programmable networks clearly facilitates service instrumentation, it does not pursue service management aspects and does not deal with customer-provider interaction, as this paper does.

3 The Concept of an Active Network

End systems attached to a network are open in the sense that they can be programmed with appropriate languages and tools. In contrast, nodes of a traditional network, such as ATM switches or IP routers, are closed, integrated systems, whose functions and interfaces are determined through (possibly lengthy) standardization processes. These nodes transport packets (or cells) between end systems. The processing of these packets inside the network is limited to operations on the packet headers, primarily for routing pur-

poses. Specifically, network nodes neither interpret nor modify the packet payloads.

Active networks break with this tradition by letting the network perform customized computation on entire packets, including their payloads. As a consequence, the active network approach opens up the possibilities of (1) computation on user data inside the network and (2) tailoring of the packet processing functions in network nodes according to service-specific requirements.

Active networks transport *active packets* [9] (also called capsules [10]). Active packets carry programs, in addition to data. A network node executes such a program, which possibly modifies the nodes' state and possibly generates further active packets to be sent over the outgoing links. Specifically, an active packet can include a program that modifies or replaces the nodes' packet processing function.

Similar to traditional networks, an active network consists of active network nodes, which are connected via links. In addition to transmission bandwidth, the key resources of an active network node include memory and CPU resources for processing active packets.

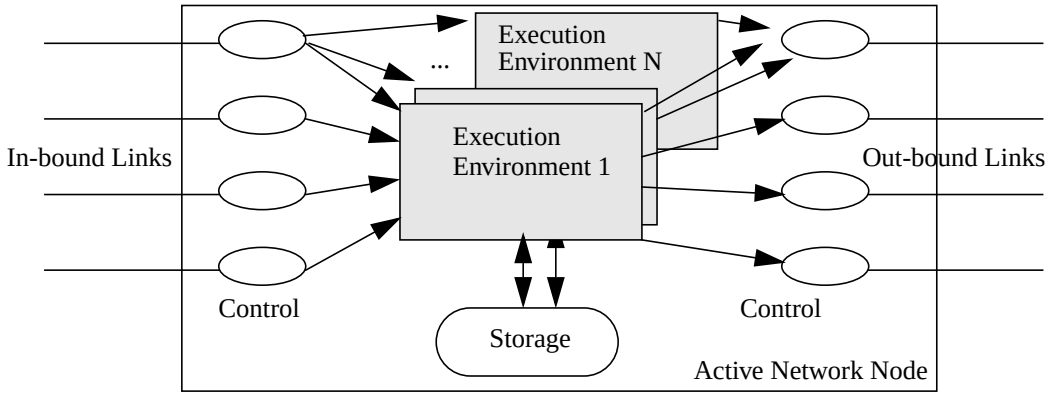

Figure 1: Architecture of an Active Network Node

Figure 1 gives a model of an active network node. The basic functions of such a node are (1) the control of the incoming packets, (2) the control of the outgoing packets, (3) packet processing, and (4) memory access. An active network node runs several execution environment in parallel. Each active packet arriving at an in-bound link contains an identifier of the execution environment that will process this packet.

Figure 1 also illustrates that an active network node can be seen as a generalization of traditional network node, such as an IP router. An IP router is limited in the sense that there is only one execution environment and a single set of pre-installed functions for packet processing.

A different view of an active network node is given in Figure 4, which stresses operating system aspects. This figure is specialized for a provider-customer environment where each customer service is run in a separate execution environment.

The networking community is currently putting substantial efforts into investigating the

active networking approach. Additional motivation for this work and ongoing projects can be found in [10]. Two areas of current research are (1) designing execution environments for active network nodes ([11], [12], [6]), and (2) extracting application-specific functionality to be integrated into the network layer, such as, application-specific packet filtering functions and application-specific packet routing ([13], [14], [15]). An architectural framework for active networks is being developed by the AN Working Group [9].

We believe that, for the active networking approach to have significant impact on the development of the telecom field, two major obstacles related to *performance* and *security* must be overcome. First, it must be possible to build active networking nodes that process packets at a rate comparable to today's IP routers or ATM switches. Second, it is crucial to engineer a secure environment, in which different parties can share communication, processing and storage resources. (This problem is currently being addressed by the AN security working group [16]).

4 Provisioning and Management of Active Network Services

4.1 Customer-Provider Interactions in Traditional Telecom Environments

Figure 2 shows the interaction taking place between a customer domain and a provider domain for the purpose of service provisioning, service delivery, and service management. Depending on the type of service, customers and providers interact in two fundamentally different ways. The first way is characterized by a provider offering

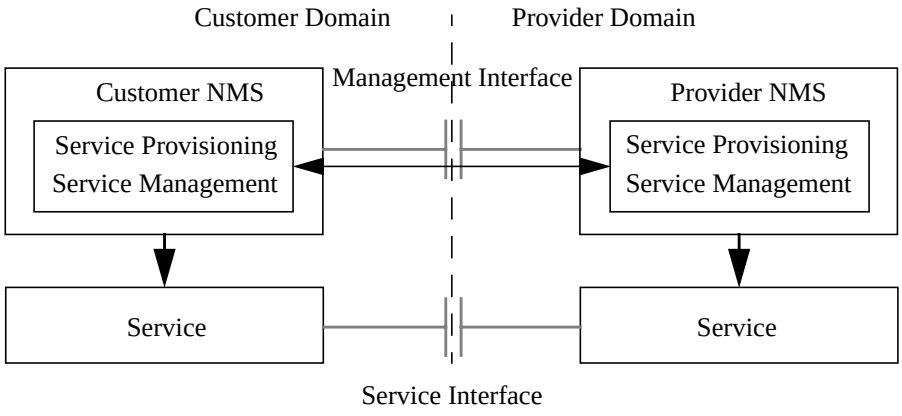

Figure 2: Management Interaction in a Traditional Telecom Environment

functionality in its domain through a service interface. A typical example of such a service is a virtual network, e.g., a Virtual Path (VP)-based service, which is purchased by a customer with geographically separated premises networks in order to construct a company-wide enterprise network. In the provisioning phase, the customer negotiates the connectivity and resources of the service, i.e., the virtual network topology and the quality of service (QoS) requirements. (A customer may request a set of constant bit rate (CBR) VPs of a certain bandwidth with delay and loss bounds.) This interaction takes place via the management interface, which interconnects the customer's and provider's management systems, and generally includes communication between human operators on both sides. The provisioned service can be accessed via the service interface, which

corresponds to the user-network interface (UNI) in our example.

The second way of interaction shown in Figure 2 relates to the case where a customer outsources control and management of a specific service to a provider, which installs and runs the service in the customer domain. In this case, the customer gives the provider access to its networking elements via management interfaces. The customer is not involved in service installation, upgrade, and management, but can concentrate on its core business instead. The provider customizes the service according to the customer's requirements.

Note that a business entity can play both the roles of customer and provider at the same time. Take the example of a Virtual Network provider, which acts as a provider vis a vis a corporate customer and as a customer vis a vis the carriers whose resources are used to build a virtual network.

4.2 Service Provisioning in Active Network Environments

In an active networking environment, the above described two ways of interactions between a customer and a provider can be realized in a flexible way with respect to service abstractions and control capabilities for the customer in the provider's domain and vice versa. In the following, we outline our framework for interaction in an active networking environment.

Figure 3 shows the interaction between a customer domain and a provider domain for service provisioning, service delivery and service management in our framework. When comparing Figure 3 with Figure 2, key differences between a traditional and an active environment become clear. First, the active networking approach allows for a clear separation between service provisioning and service management. Second, service management can be realized via the service interface.

Service provisioning includes creating service functionality and securing resources for the service. It is performed in a cooperative manner between the customer and the provider through a management interface. The task of provisioning can continue during the operational phase of a service, e.g., for the purpose of renegotiating bandwidth associated with a virtual network.

In a traditional telecom environment, the process of service provisioning and the resulting service abstractions are service-specific, whereas in an active networking environment, both the provisioning process and the service abstraction can be realized to be truly generic. Provisioning, say, a virtual network service in a traditional telecom environment, includes setting up virtual links between customer premises networks and allocating resources to these virtual links. For the customer, the service abstraction consists of a set of links, associated with bandwidth and QoS guarantees.

In contrast, service provisioning in our framework for an active telecom environment gives the customer a more complex, but also more powerful service abstraction. This abstraction consists of a graph of virtual active nodes interconnected by virtual links. During the provisioning phase, the customer and the provider cooperatively set up such a graph and allocate resources to the service. These resources include bandwidth for the

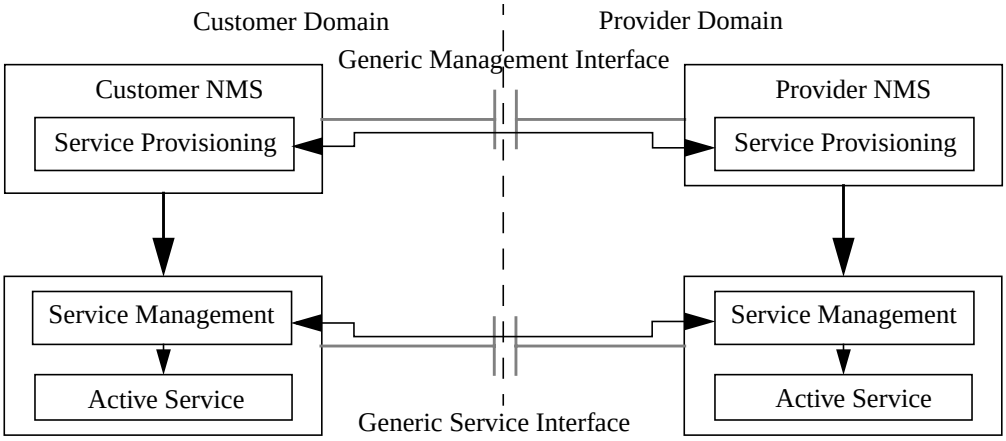

Figure 3: Management Interaction in an Active Telecom Environment

virtual links, as well as processing resources and memory for the virtual active nodes. We call this service abstraction a *Virtual Active Network (VAN)*. (The term Virtual Active Network is also used by other authors in a slightly different way [6].) The Virtual Active Network is a generic abstraction in the sense that it provides the customer with a platform for running a large class of services, not just a single network service. (Note the difference between the VAN, which generic, and the virtual network in the previous paragraph, which is a specific service.) It is a powerful abstraction in the sense that the customer can, within the limitations of the specific VAN topology and the allocated resources, install, configure and run network services without further interaction with the provider.

A example is discussed in Section 5, where a customer installs a set of virtual routers and links in a provider domain.

4.3 Service Management in Active Network Environments

Service management, as understood in this paper, consists of the creation, supervision, updating and removal of a service on a networking platform. While the process of provisioning a Virtual Active Network has to be performed in cooperation with the provider via the management interface, the task of managing the particular service a customer wants to run on a Virtual Active Network can be performed via the service interface and, therefore, without interacting with the provider's management system. Figure 3 illustrates this aspect.

We outline how the service management task is realized with respect to the two types of customer-provider interaction described in Section 4.1.

In traditional network environments,--as the virtual network example shows--the customer has a simple service abstraction and sees the service essentially as a black box with access points. Service management functionality is supported by the provider through the management interface (Figure 2) and is limited by the simplicity of the service abstraction, which generally supports only monitoring functions.

In an active network environment, a customer can install service delivery and service

management functionality on the VAN purchased from the provider. The detailed service abstraction a VAN gives allows for control on a very detailed level, down to changing the packet scheduling policies in the active nodes. Examples of customer controlled management functions inside the provider's domain include installing and monitoring virtual routers, updating an existing service, performing load balancing operations and creating end-to-end connections with QoS guarantees inside a VAN. These examples are described in more detail in Section 5.

For a customer, there is a trade-off between increasing the controllability of a service and increasing the complexity of the customer's control and management system. For instance, an end-to-end link is relatively simple to manage, but it is also a less controllable service abstraction than a graph of fully configurable virtual network nodes. Therefore, a designer of an active network service has to engineer a service abstraction with the degree of controllability that meets best the customer's requirements.

In the case where a customer, having an active networking infrastructure on its premises, outsources a specific service, a provider can install, configure, and update this service through the service interface shown in Figure 3.

4.4 Architecture of an Active Network Node in a Telecom Environment

Figure 4 gives an operating system point of view of an active network node in a telecom environment. A node operating system layer configures and provides access to the node's resources, such as links, processing and memory resources. This layer runs the execution environments, separates them from each other, and polices the use of the resources consumed by each execution environment.

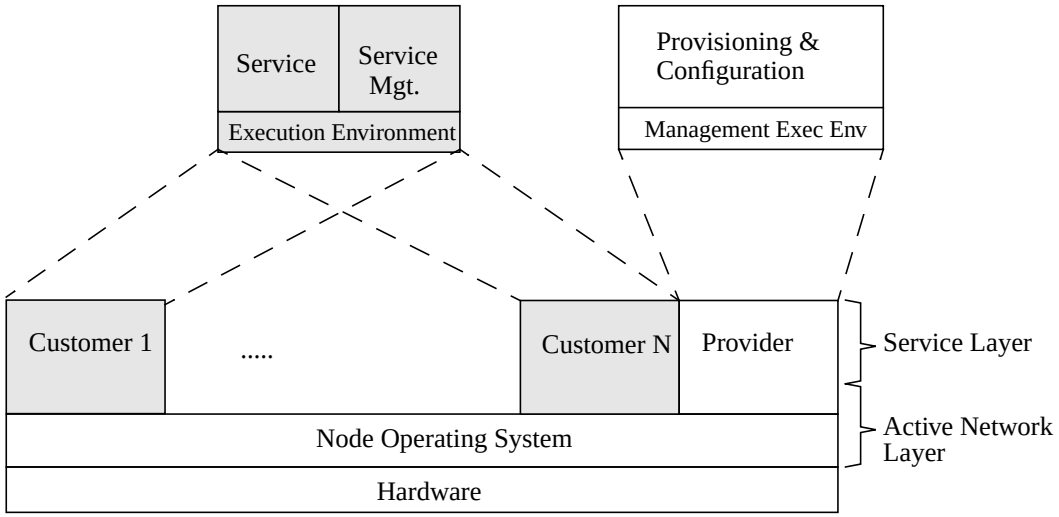

Figure 4: Architecture of an Active Network Node in a Telecom Environment

Figure 4 specifically shows the case where a provider offers Virtual Active Networks to a several customers (Customer 1,..., Customer N). The figure shows one node of such a VAN. Each customer runs its service in a separate execution environment. A special execution environment runs the provider's VAN provisioning and configuration system,

which creates execution environments for customers and is able to modify and terminate them. The VAN provisioning and configuration system is controlled by the provider's management system via the exchange of active packets. The provider's management system, as shown in Figure 3, maintains a global view of the provider's domain for the purpose of VAN provisioning.

During the VAN provisioning phase, execution environments are created on a set of active network nodes, according to a customer's requirements. These execution environments are connected via virtual links; they collectively form the VAN for this particular customer. Without further interaction with the provider's management systems, the customer configures its execution environments for the specific service it wants to run and for the service management functionality it needs. This configuration process is realized by the customer sending active packets, which contain the code for customizing the execution environments, to the nodes of the VAN. (See Section 6 for a specific example of such a configuration.)

In the case where a customer outsources network services, the node architecture shown in Figure 4 is also applicable. In this case, a customer installs a network of active nodes on its premises and purchases services from one or more providers. A provider installs, runs, and manages the purchased service on the customer network. The customer controls the node operating systems and the provisioning environment and creates an execution environments for each service it wants to purchase.

5 Management Scenarios in an Active Telecom Environment

This section attempts to illustrate the concepts introduced in Section 4 with some sample scenarios. In most of the scenarios, a customer installs "virtual" routers in the provider's domain. While an Intranet based on virtual routers might not be the most popular service to be installed in a future active environment, we chose it, because it is easy to understand and it illustrates the powerful capabilities that an active environment can provide.

5.1 Customer Installs an Internet Virtual Private Network in the Provider's Domain

A customer wants to interconnect geographically separated premises networks that run an IP service, in order to create a company-wide Intranet.

Today, there are several solutions for interconnecting customer networks using a public service provider. Customer networks are often interconnected via leased lines, or, more recently, by links based on ATM or SONET technology. These links generally have specified quality of service requirements, which must be met by the provider. Another solution for interconnecting customer Internets is based on "tunneling" packets over a provider's Internet infrastructure. In all these cases, the service abstraction can be described as a set of links between the edge routers of the customer premises. This service abstraction is created by the provider during the provisioning phase and is the basis for the service contract between customer and provider.

With underlying active networking technology in a provider's domain, a customer can lease a Virtual Active Network from a provider. On such a VAN, the customer can install an IP service by configuring and running virtual routers, i.e., routers consisting entirely

of software components, on the active nodes of the VAN. Such a scenario allows the customer to manage the virtual routers in the provider's domain in the same way as the (hardware) routers in its own domain, without interacting with the provider's management system. This is achieved by installing and running the virtual routers including their management functions in the execution environment of the active network nodes of the leased VAN (Figure 4), and by communicating with active packets via the generic service interface (Figure 3).

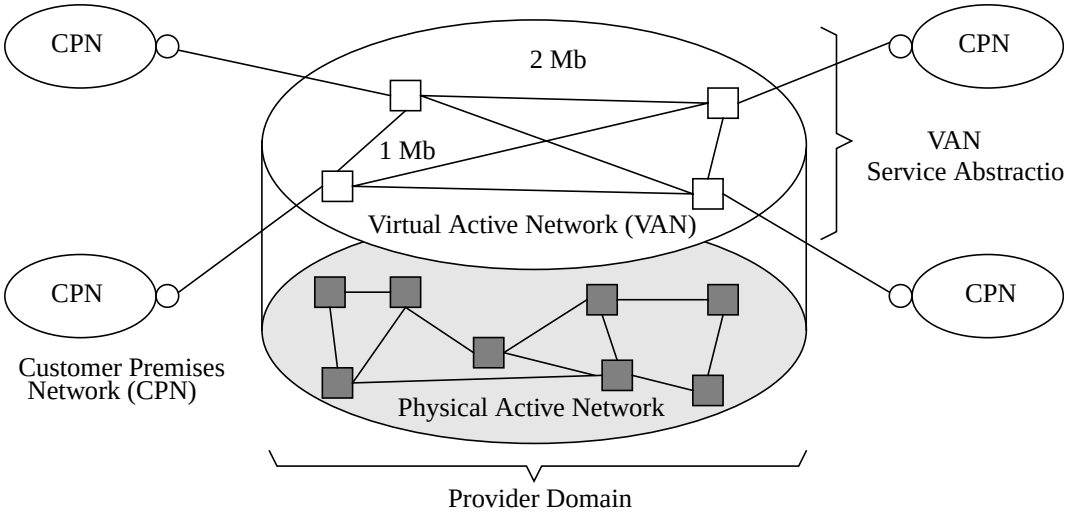

Figure 5: The Virtual Active Network as the generic Service Abstraction

5.2 Customer Performs Load Balancing within the Virtual Network

The detailed service abstraction of a VAN in the above case, which gives the customer access to nodes inside the provider's domain, allows the customer to perform load balancing operation on the Internet virtual private network, which would not be possible with the traditional, less complex service abstraction of a virtual private network as a set of end-to-end links.

5.3 Customer Sets up an Internet "Virtual Path"

By having access to the virtual routers introduced above, a customer can partition the output buffers of these routers, in order to allocate resources to a specific traffic class. (We assume here a multiclass Internet service, following, e.g., the Differentiated Services concept [17].) By doing so, a customer can install a virtual path through the virtual private network, and can provide QoS guarantees on this virtual path--within the limits of the VAN resources--according to its own specific requirements.

5.4 Provider Upgrades a Service in the Customer Domain

A customer with an active networking infrastructure can outsource a specific service to a provider. This provider installs, runs and supervises the service in the customer domain. An upgrade to this service can be performed by the provider through the service interface in Figure 3. Since an active network node runs several execution environments concurrently, the service provider can install a new version of a service without interrupting the current version. Furthermore, the provider has the option to run different versions of a

service at the same time on the customer's infrastructure.

6 A Platform for Evaluating and Experimenting with AN Concepts

We are in the process of building an active networking platform, in order to test, evaluate and demonstrate active networking services and management concepts. The core of this platform consists of a cluster of Ultra-SPARCs, interconnected via an Ethernet and an ATM LAN. The workstations run the active network nodes (each node on a separate workstation), the traffic generators used to inject active packets into the network, and a management system that allows to monitor and control the active network nodes.

All software components of our platform are written in Java. We chose Java because of its strengths as a prototyping language for a networking environment, and because Java directly supports the realization of active messages through the concept of mobile code. In our implementation, an active network node is implemented entirely in software, which gives us the flexibility for experimenting with different designs.

In the reminder of this section, we describe the realization of an active network node on our platform. We show, how our implementation supports the creation of an execution environment, the installation of a specific service, and the upgrade of such a service.

6.1 Realizing an Active Network Node

The active network node architecture given in Figure 1 and Figure 4 is realized as follows. All the links, the node operating system, and the execution environments run as separate threads. The in-bound links deliver the incoming packets to the appropriate execution environments, according to an identifier in the packet header. The node operation system schedules the execution environments, taking into account to the processing resources allocated to the execution environments. The out-bound links run the packet schedulers on the outgoing multiplexers and transmit the packets produced by the execution environments to neighboring nodes.

The complexity of the active network node we have built is currently in the order of 150 Java classes with 7000 lines of code.

6.2 Managing an Active Network Node

We have implemented a system for a human operator to manage an active network node. The system currently has the following functionality. First, it allows us to set up and configure execution environments and allocate resources to these environments (see Section 6.3). This functionality is needed for VAN provisioning. Second, the system allows us to install cut-through links as part of the VAN provisioning process. Third, the management system enables us to modify the policies of the scheduler that controls the processing resources of the execution environments and the scheduler that gives the outgoing buffers access to the link bandwidth. (The system currently supports two policies: a round robin policy for best effort services and a fair sharing policy with QoS guarantees for guaranteed services.)

6.3 Realizing an Execution Environment for a Customer Service

When our management system creates an execution environment in an active network node, a structure shown in Figure 6 is initialized. This structure represents a node in a VAN. The execution environment contains virtual in-bound and out-bound links (each of which corresponds to a link of the active network node), a central queue for incoming packets, a packet processing element, and storage for data and functions. The packet processing element takes active packets out of the central queue and processes them, using the basic instruction set of the execution environment, which is a subset of Java, augmented by service primitives, e.g., for storage access.

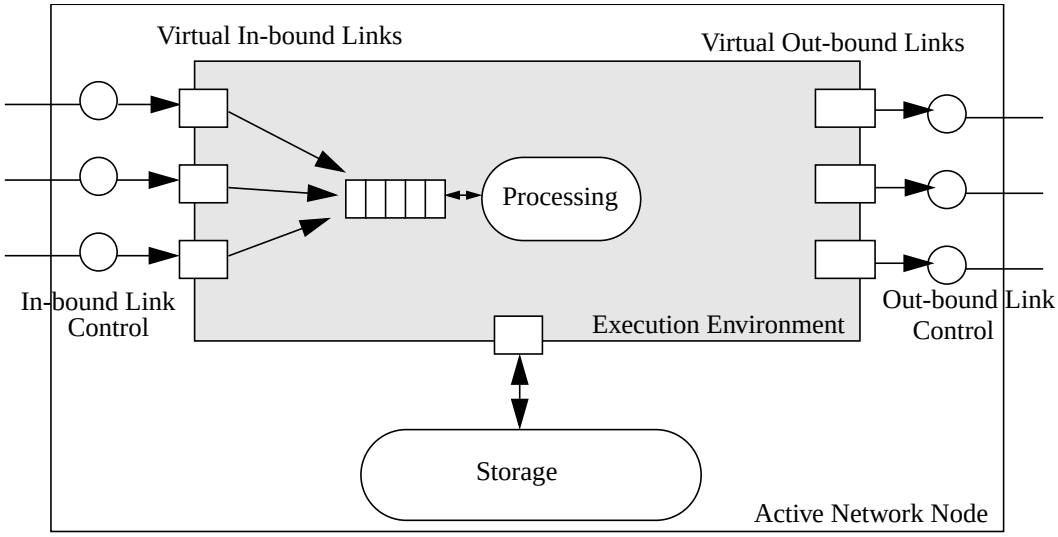

Figure 6: Initial Configuration of an Execution Environment

When creating an execution environment, processing resources are allocated to this environment. In our implementation, this is realized by creating a thread that runs the packet processing element and by registering this thread with the scheduler of the node operating system.

Figure 6 can be interpreted as a virtual machine. It allows us to install and configure a service-specific, more complex virtual machine, by sending active messages that contain the code for the new virtual machine to the execution environment. This process can be compared to boot-strapping an operating system on a computer. Using this method, a customer can install a specific network service in an execution environment that has been created by a VAN provider.

6.4 Installing and Running a Multiclass IP Service

Installing an IP service on a active network node is achieved by configuring a virtual router inside the execution environment shown in Figure 6. The result of this process is the environment displayed in Figure 7. The service installation is triggered by sending a sequence of active packets to the execution environment. Processing these packets one by one results in installing an IP routing table, creating output buffers for the virtual outbound links, setting up packet schedulers that operate on these buffers, installing func-

tion code for routing and management operations, configuring service-specific control parameters and management parameters, etc. After that, in a similar way, the IP service is initialized, which includes starting the routing protocol.

IP packets arriving at an in-bound link of this initialized execution environment are forwarded by the processing function according to the routing table and the class identifier in the packet header. A packet with the destination address of this virtual router, triggers the execution of a specific local function, such as a routing table update or a management operation.

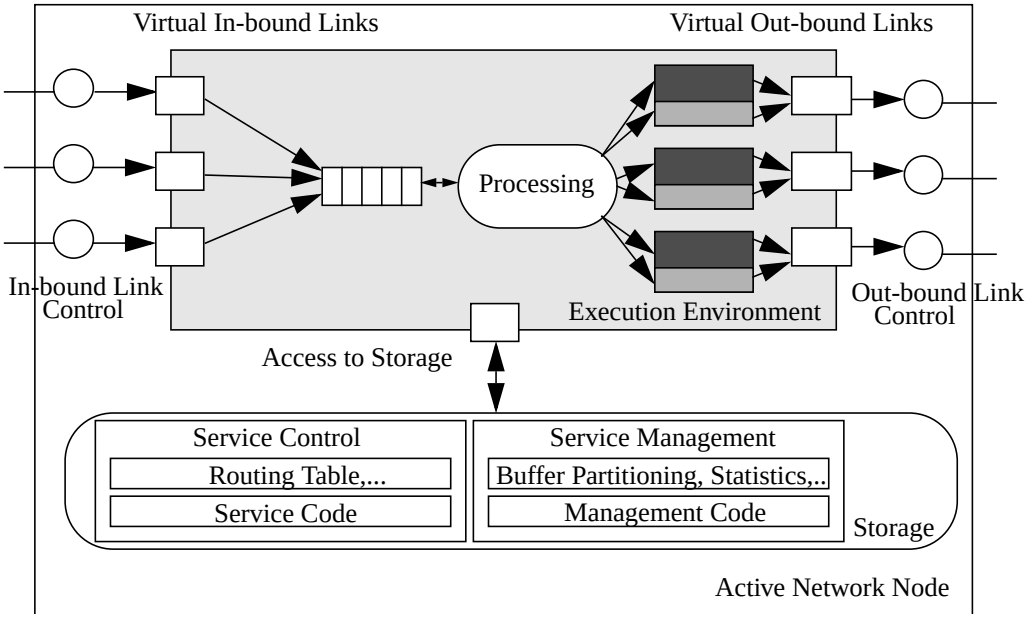

Figure 7: Installing an IP Service in a Execution Environment

In our prototype implementation, we can change the partitioning of the output buffers and can monitor the buffer usage by sending active packets to the virtual routers installed on the platform. This way, we can perform service management operations, as a customer would do while managing its IP service on a VAN.

6.5 Upgrading the IP Service

On our testbed, we can demonstrate upgrading the above described IP service to a service with multicast capabilities. The process of upgrading a service follows the same scheme as the process of initializing a service described in 6.4. The virtual machine given in Figure 7 must be reconfigured, primarily by adding new objects and functions for multicast routing and for the management of multicast addresses. We can perform this service upgrade on an active node without interrupting the running IP service. This is possible, because the new service is an extension of the running IP service, and the upgrade process is performed as a sequence of atomic operations, each of which leaves the node in a consistent state, as far as the running IP service is concerned.

7 Discussion

In this paper, we have investigated the problem of service provisioning and service management in a telecom environment that is based on active networking technology. Starting from the facts that an active network node is able to run different execution environments concurrently, and that it can be accessed via a generic service interface, we have explored the possibilities for service provisioning and service management, primarily with respect to customer-provider interactions. Based on this work, we conclude that with the introduction of active networking technology a range of new and flexible options emerge for supporting management tasks.

The paper includes the following contributions. We have outlined a management framework for a telecom environment with the benefits of

- supporting generic, i.e., service-independent, interfaces for service provisioning and management;
- giving customers the option to manage their service in the provider's domain without interacting with the provider's management system;
- giving providers the possibility to configure service abstractions with different degrees of detail and controllability, according to a customer's requirements;
- allowing a customer to accesses a service in the provider's domain, as well as to outsources a service to a provider.

Further, we have illustrated the above framework with a series of scenarios, and we have described the realization of some key functions of this framework on our active networking platform.

All of the above listed properties are consequences of having an underlying active network, and they are much harder if at all to support with traditional networking technology. Note, however, that active networking is a field still in its infancy. This paper makes the assumption that active networking technology will mature and that it will become commercially successful. As we have pointed out, the major obstacles on this road relate to performance of active network nodes and security concerns. Both of these issues are currently being addressed by the active networking community.

Our work to date opens up the way for further research. From the management perspective, some of the topics worth pursuing are:

- In Section 4, we have informally introduced the Virtual Active Network (VAN) as the generic service abstraction in an active telecom environment. This concept needs to be formally defined, together with the VAN provisioning interface.
- In Section 4, we have outlined an architecture of an active network node. When realizing this architecture, functional support has to be provided within the node operating system and the management execution environment to partition the node resources and to police their access by the customers.

- A methodology and tool-kits need to be developed to support the design and implementation of active services and their management. A possible candidate for such a methodology is a mobile agent paradigm.

8 References

[1] M. C. Chan, A. A. Lazar, R. Stadler, "Customer Management and Control of Broadband VPN Services," Fifth IFIP/IEEE International Symposium on Integrated Network Management, San Diego, California, U.S.A., May, 1997, pp. 301-314.

[2] Y. Yemini, G. Goldszmidt, S. Yemini, "Network Management by Delegation," Second International Symposium on Integrated Network Management, Washington DC, April 1991, pp. 95-107.

[3] A. Vassila, G. Pavlou, G. Knight, "Active Objects in TMN," Fifth IFIP/IEEE International Symposium on Integrated Network Management, San Diego, California, U.S.A., May, 1997, pp. 139-150.

[4] Y. Kiriha, M. Suzuki, I. Yoda, K. Yata, S. Nakai, "Active Q Adaptor: A Programmable Management System Integrator for TMN," Distributed Systems and Operations Management (DSOM98), 1998.

[5] Y. Yemini., S. da Silva, "Towards Programmable Networks," IFIP/IEEE International Workshop on Distributed Systems: Operations and Management", L'Aquila, Italy, October 1996.

[6] S. Da Silva, Y. Yemini, D.Florissi, "The Netscript Project," ICC Workshop on Active Networking and Programmable Networks, Atlanta, 1998.

[7] J. E. van der Merwe, I. M. Leslie, "Switchlets a Dynamic Virtual ATM Networks," Fifth IFIP/IEEE International Symposium on Integrated Network Management, San Diego, California, U.S.A., May, 1997, pp. 355-368.

[8] A.A. Lazar, K.S. Lim, F. Marconcini, "Realizing a Foundation for Programmability of ATM Networks with the Binding Architecture," IEEE Journal of Selected Areas in Communications, Special Issue on Distributed Multimedia Systems, Vol. 14, No. 7, September 1996.

[9] AN Architecture Working Group, "Architectural Framework for Active Networks," K. Calvert (editor), 1998.

[10] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Weatherall, G. J. Minden, "A Survey of Active Network Research," IEEE Communications Magazine, Vol. 35(1), pp. 80-86, 1997.

[11] D.J. Weatherall, J.V. Guttag, D. L. Tennenhouse, "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols," IEEE OPENARCH, San Francisco, CA, April 1998.

[12] J. Smith, D. Fraber, C. Gunter, S. Nettles, D. Feldmeier, W. Sincoskie, "SwitchWare: Accelerating Network Evolution," Technical Report MC-CIS-96-38, CIS Department, University of Pennsylvania, May 1996.

[13] S. Bhattacharjee, K. L. Calvert, E. W. Zegura, "An Architecture for Active Networking", in proceedings of High Performance Networking, 1997.

[14] U. Legedza, J. Guttag, "Using Network-level Support to Improve Cache Routing", 3rd International WWW Caching Workshop, Manchester, England, June 1998.

[15] S. Bhattacharjee, K. L. Calvert, E. W. Zegura, "Self-organizing wide-area network caches," IEEE INFOCOM, 1998.

[16] AN Security Working Group, "Security Architecture Draft," S. Murphy (editor), 1998.

[17] W. Feng, D. Kandlur, D. Saha, K. Shin, "Adaptive Packet Marking for Providing Differentiated Services in the Internet," International Conference on Network Protocols ICNP, October 1998.