Programming Telecommunication Networks

Aurel A. Lazar
Columbia University

Abstract

The recent move toward market deregulation and open competition has sparked a wave of serious introspection in the telecommunications service industry. Telecom providers and operators are now required to open up their primary revenue channels to competing industries. The competition for product differentiation increasingly depends on the level of sophistication, degree of flexibility, and speed of deployment of services that a future provider can offer. These factors in turn depend heavily on the flexibility of the software architecture in place in a provider's operational infrastructure. Within this context, we examine the service architecture of two major global communication networks — the Telephone Network and the Internet and explore their weaknesses and strengths. We discuss the realization of an *open programmable networking* environment based on a new service architecture for advanced telecommunication services that overcomes the limitations of the existing networks. Our approach to network programmability stems from two angles — one conceptual, the other implementational. In the first, we attempt to develop a service model that is open and reflects the economic market structure of the future telecommunications service industry. We believe that investigating such a model will help clarify some of the pertinent issues confronting the telecommunications service industry today as it comes of age. Furthermore, we introduce an extended reference model for realizing the service marketplace and present it as a vehicle for creating multimedia services with QoS guarantees. In the second, we investigate the feasibility of engineering the reference model from an implementation standpoint. We describe a realization of the open programmable networking environment as a broadband kernel. Called xbind, the *broadband kernel* incorporates IP and CORBA technologies for signaling, management, and service creation, and ATM for transport. We address some of the important QoS, performance, scalability, and implementation issues, fully aware that our work has opened new vistas that call for additional research.

The ability to rapidly create and deploy new and novel services in response to market demands will be the key factor in determining the success of the future service provider. As the high-speed switching and communication infrastructures improve and bandwidth becomes a commodity, it is envisioned that the competition for product differentiation will increasingly depend on the level of sophistication, degree of flexibility, and speed of deployment of services that a future provider can offer [1]. These factors in turn depend heavily on the flexibility of the software architecture in place in a provider's operational infrastructure.

The current generation of telecommunication networks is based on an architecture over 30 years in age [2]. The basic tenet behind the architecture is the implicit assumption that Customer Premises Equipment (CPE) has no computational capabilities and limited modes of interaction. This assumption eventually translated into a design somewhat akin to a mainframe cluster model where a small number of computationally powerful processors called Service Control Points (SCPs) are distributed throughout the network and take on the responsibility of service provisioning for all connected CPEs. As in the mainframe cluster model, the CPEs themselves act solely as the user interface channeling simple user requests and responses into the system. Within the network, dedicated processors running specialized monolithic software optimized for efficiency process, coordinate and translate these requests and responses into the necessary data and connections that constitute the service.

The primary deficiency of this architecture is its monolithic view of the service provisioning process. The network operator assumes almost complete control over the decisions pertaining to the design, introduction, and management of services since it owns the SCPs. Interfaces to the service man-

This article is a revised version of the keynote address given by the author at the International Workshop on Quality of Service, Columbia University, New York, NY, May 21–23, 1997. Published in the workshop proceedings.

agement infrastructure are often nonexistent, proprietary or narrow in scope and intimately coupled to the hardware they operate on. As a result, any third-party involvement in service programming is limited to customizing only a small set of operational parameters. Furthermore, deploying new services in today's telephone networks takes up to several years primarily because the software systems with which they need to be integrated are enormously complex and prone to many cross-service interactions.

IP based networks like the Internet are known for their scalability and resilience to partial failures. Their reliance on datagram forwarding on a hop-by-hop basis makes them ideal for the transport of short control messages with little or no call holding times. Connection-oriented broadband networks based on asynchronous transfer mode (ATM) technology permit a much higher degree of predictability to be engineered relatively simply, because of their inherent resource partitioning capability. They are thus highly suited for transport of streams with quality of service (QOS) requirements and long call holding times.

The capabilities of modern computer systems has advanced well beyond the stifling limitations imposed by these early architectures. Modern software engineering has advanced to the point where industry standards [3] now exist for implementing platform independent distributed component based software. These and newer emerging standards allow the construction and packaging of independent software components into suites which can further be assembled via application-level frameworks to create a truly distributed information infrastructure on a global scale. While these modern software engineering aids by themselves do not solve the fundamental problems inherent in any scalable distributed system, they do provide an excellent infrastructure for dealing with problems of programmability, portability, maintainability and reusability, problems frequently faced by the telecommunications software industry.

Our approach to these problems stems from two angles one conceptual, the other implementational. In the first, we attempt to develop a service model that is open and reflects the economic market structure of the future telecommunications service industry. We believe such a model will help clarify some of the pertinent issues plaguing the telecommunications service industry today as it comes of age. In the second, we investigate the feasibility of engineering this model as an open programmable networking environment. We address some of the important implementation issues fully aware that our work has opened new vistas that call for additional research.

We discuss the realization of the open programmable networking environment as a broadband kernel. Called *xbind* [4, 5], the broadband kernel is a programmable operating platform that supports the creation, deployment and management of networked multimedia services (e.g., virtual circuits, virtual paths, virtual networks, multicast, etc.) and mechanisms for efficient resource allocation (e.g., connection management, route management, admission control, QoS mapping, etc.). The term "kernel" is deliberately used to draw a parallel between its role as a resource allocator and extended machine, and that of a typical operating system. The broadband kernel behaves as a resource allocator because it mediates and arbitrates between conflicting requests for resources made by various parties in the system. It functions like an extended machine because it provides a simplified means of accessing fundamental system services by abstracting away the operational complexities of provisioning these services.

xbind exploits the advantages offered by IP and ATM technologies without necessarily suffering their shortcomings. This is based on the observation that while typical inter-object communication modes are mostly restricted to exchanges of short messages, the transport of multimedia streams involve typical call holding times several orders of magnitudes longer in duration. Furthermore, while inter-object communication requires reliability even under adverse partial failure conditions, multimedia stream transport requires strict guarantees on delay bounds and losses. In this respect, it seems almost natural to use IP for inter-object communication (which we term signaling) while ATM for media stream transport.

We begin the second section by proposing a simple classification scheme for services. We use this scheme to examine the service structure of two prominent networks — the Telephone Network and the Internet. In the third section, we introduce the basis of an economic model for describing future telecommunication services based on market forces. We believe this will be the end result of a natural evolution of the industry given the current trends in deregulation and open competition. In the fourth section we propose a model for a service architecture based on the principles of open APIs that closely parallels our economic model and briefly list out some of its principle components. In the fifth and sixth sections we present the service creation and the resource allocation model, respectively. The engineering aspects of architecting the xbind broadband kernel including QoS, performance, scaling, and implementation issues are dealt with in the seventh section. Concluding remarks are presented in the final section.

Service Architectures: A Taxonomy of the State of the Art

A service architecture defines the structure and mode of operation of a facility that offers a service. There are two key categories of services in most communication infrastructures:

Basic Communication Services — which focus on the mechanisms and the interactions between network entities so as to enable the communication process

Content Services — which focus on the means of access, presentation, and organization of content resources in the network to facilitate communication.

A third category of services deals with value-added enhancements over the basic communication services. Conceptually these services lie between the two basic categories described above in the sense that their utility is dependent on the functioning of the basic communication service yet by themselves are not considered as such. These services are exemplified by the Advanced Intelligent Network (AIN) [6] services and range from convenience features like call forwarding and caller ID, to more complex services like mass calling.

Service Requirements; The Need for Domains

The conflicting forces of market demand for flexibility, accountability, and robustness makes the task of architecting an open service model a difficult technical challenge. Basic communication services lie at the base of the service spectrum supporting the fundamental mechanism for information exchange upon which the other service categories are built. Thus, the primary requirement of basic communication services is one of robustness, high availability, and low failure rate. Content services, on the other hand, lie at the opposite end of the spectrum. Market forces demand these to be highly customized, user-oriented, and easy to deploy.

Hence, as we ascend the hierarchy of services from the basic communication services to sophisticated content-driven end-user services, the need for programmability also rises. The difference in requirements for these service categories often result in different technical solutions being employed for addressing the issues pertinent to each category. In effect these enforce a natural domain-like separation between network concerns, service concerns and user concerns. A similar view is reflected in the Telecommunications Information Networking Architecture Consortium (TINAC) stakeholder domains [7] which classify TINAC services into network provider domains, service provider domains and user domains.

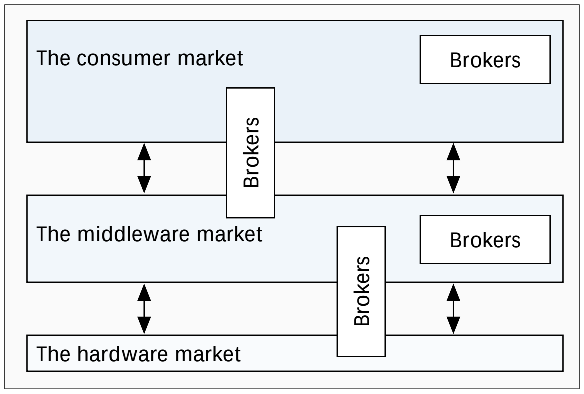


Figure 1. *The layered market model.*

In the Internet there is no strict distinction between network provider, content provider, and user [8]. In practice, domain-like separation (usually for security and administrative reasons) is typically imposed through artificial means (like addressing structure and naming). In this model, any user can also be:

- A network provider by achieving physical connectivity with the Internet and offering connectivity services to other users.
- A content provider by making available content for public access and advertising its availability through a directory service.

Service Models: A Question of APIs?

The service architectures of the Telephone Network and the Internet differ in several aspects. In the Telephone Network, historical assumptions about CPE capabilities have led to a two tiered architecture consisting of a user domain and a network domain. (The service architecture of ATM Networks is closely mirroring the one of the Telephone Network.) Users lie at the periphery of the system and access network services via a thin interface known as the User Network Interface (UNI) [9] while within the network domain, interconnection is achieved via a complex Network Node Interface (NNI). In this model, there is little distinction between network provider and service provider, since most useful services (in particular, Intelligent Network (IN) types) require the intimate network support available only through an NNI. In this sense, although there is a clear separation between bearer services and AIN services in a technical and engineering sense, in reality, until only recently the administrative, operational, and business concerns of an enterprise often make it more lucrative to merge network and service provider roles as one. For instance, in the past, protective regulatory structures had always afforded network operators the luxury of using AIN services as a value-added component to enhance the marketing and sales of plain old bearer services.

The recent 1996 Telecommunications Act, however, has demolished to a considerable extent many of these protectionist measures, paving the way for freer competition at all levels in the market. One of the more significant changes mandated by the new regulations effectively require that network operators provably demonstrate capability to support third party service provisioning. These and other emerging trends are indicative of the need for a serious reexamination of current telecommunication service models (and the ensuing business practices) or risk serious financial consequences.

The intimate coupling of the communications and service architecture makes the introduction of new services, particularly those that do not conform to the traditional point-to-point connection paradigms, clumsy and restrictive because:

- The interface between the network and the service architecture responsible for basic communication-services-like connection setup procedures) is rigidly defined and cannot be replaced, modified, nor supplemented — all services must be implemented in terms of these.
- The interface between individual services is defined by the rather restrictive Intelligent Network Access Point (INAP) protocol which all conceivable service-level signaling procedures and semantics must map into.

An even worst situation presents itself if we examine the boundary between the network and the user. The potential diversity and flexibility requirements of user level services and applications far exceed that of typical AIN services demanding all the more an open environment for design, installation, and operation. The simple UNI is difficult to extend and was never designed for these requirements.

In terms of APIs for content provisioning, the Internet far excels the Telephone or ATM Networks. The Java virtual machine [10] in effect can be seen as a freely extendible API for content services whose basic parameters are programs instead of simple types. Even at the basic communication services level, IP options provide, in principle, a primitive API for influencing basic packet routing policies. In reality, most of these APIs are usually not fully implemented or supported. For example, the source-routing option of IPv4 is an example of an API which, if fully supported by all IPcapable hosts, would provide an elegant solution to the problem of host mobility without the need for tunneling.

Support for Quality of Service

There are many commonalities between approaches to guarantee QoS in the Internet and ATM networks. The key QoS concerns in telephone networks represent a subset of QoS issues in ATM networks. For example, the Integrated Services Architecture [8] defines the notion of flows, a notion closely associated with virtual circuits in ATM networks. QoS requirements, on the other hand, associated respectively with flows and virtual circuits, seem to be widely apart. Although they both attempt to codify the predictable statistical characteristics of multimedia streams, the traffic assumptions about these continue to differ considerably. The key challenge in guaranteeing QoS is in finding the appropriate trade-off between transport efficiency and the complexity of the network control architecture.

To better understand the tradeoff in QoS support, let us consider a tandem link of an ATM network consisting of three switches (Fig. 6). Each switch accepts traffic from a set of input links, and routes each incoming cell through a non-blocking switch fabric to the appropriate output port, where it is queued in a link control unit for transmission over the output link. This link control unit is essentially a multiplexer. It consists of a set of buffers, a buffer manager, and a scheduler, and it mediates the contention among cells from different input links and among those of different traffic classes. Calls of each class are distinguished by a particular cell interarrival time distribution and a particular set of QoS requirements.

The key question is: how many calls of each class may be simultaneously supported while guaranteeing the QoS for all classes? The answer to this question comprises the definition of the schedulable region originally explored in [11]. The capacity of the link, the size of the buffer, and the scheduling algorithm used will determine the size of the schedulable region. Note that the schedulable region can be approximated by assigning an "equivalent bandwidth" [12] to each of the streams feeding the multiplexer.

From the point of view of admission control, the schedulable region is a sufficient representation of the link as it summarizes the net effect of all cell-level details. The separation between scheduling and admission control emerged as the key principle for guaranteeing QoS on multiple time scales in broadband networks [13]. Admission control calls for resource reservation mechanisms to be implemented in switches in order to provide end-to-end QoS guarantees to virtual circuits. This requires virtual circuit specific state information to be stored in switches thereby increasing the complexity of the network architecture. Similar conclusions are arrived at in the context of the Internet [8, 14].

In summary, the service model of the Telephone and ATM Networks is one of provider and customer. This is a clear demarcation in terms of the technology employed (as reflected by the APIs) and responsibility between the two distinct roles. In the Internet this distinction is less clear and a peer-to-peer model is perhaps a more accurate analogy, since users and providers both run on essentially the same technology. The obvious advantage of a peer-to-peer service model is one of flexibility since there are no technical barriers from preventing any user from setting up his/her own service. On the downside, QoS, security, policy, and standards are harder to enforce since they require cooperation from a much larger community. Finally, let us note that network architectures that guarantee QoS must control resources explicitly. Thus, this requirement should be made explicit at the fundamental modeling level.

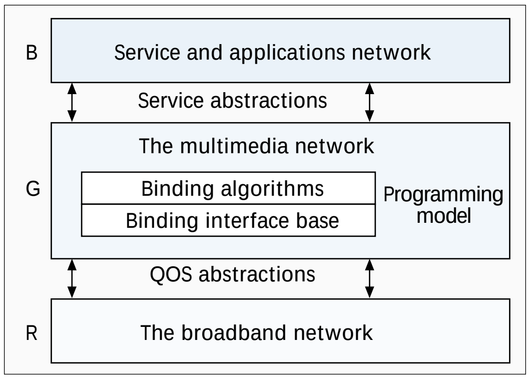


Figure 2. *Overview of the RGB decomposition of the XRM.*

Moving Beyond the State-of-Art: A Layered Market Model

In the previous section, we examined two primary service models and their respective trade-offs. We note that a key factor influencing the service model of a network depends to a large extent on the types and levels of APIs available. In fact, we believe that APIs primarily reflect the flexibility and maturity of a service architecture and good APIs at the service level are the macroscopic counterparts of good coding practices at the implementation level.

In this section, we propose a three-layered API-based service model inspired by the principle of the open market which we believe better reflects the operating structure of the future communication services industry. The basic tenet of the model is the premise that the future telecommunication service industry will operate within an open market where sufficient alternatives exist to allow services to be traded as commodities. This premise is not unreasonable given the recent trend towards deregulation and the general move towards streamlined horizontal market structures.

The model is divided into three layers with each layer representing a market. The lowest layer is a hardware market where numerous equipment manufacturers and vendors offer hardware and firmware solutions for building the basic communication infrastructure. The customers of this market are typically network carriers, third-party software developers who specialize in developing software for service providers and a handful of service providers themselves. The APIs provided by vendors in this market allow their users to write basic communication services and the associated middleware components. In the second layer is a middleware service market where carriers, software developers, and middleware service providers offer middleware service products to customers who are in the user service provisioning business. The APIs provided in this market are sufficiently high to allow development of any consumer level service. Finally, at the highest layer is the consumer services market where consumer service providers compete to bundle, integrate, and customize their wares in the most appealing form for the mass market. Within each market there may exist brokers (as rightly recognized by the TINAC stakeholder domain model [15]) whose role is to mediate dealings between buyers and sellers who, because of regulatory or business policies, cannot transact directly. The model is shown in Fig. 1.

The service model just described falls somewhere between the Internet's peer-to-peer model and the Telephone Network's strict provider-customer model. It allows, in principle, the cooperation of any number of entities in the network for realizing a common service as well as the competition among services for network resources. As will be shown in the next section, the corresponding engineering model can be parametrized in such a way that the basic characteristics of the peer-to-peer model as well as the characteristics of the provider and costumer model can be recovered. Within each layer (or market), players are free to enter and buy, sell or rebundle each other's services. Across layers, the relationship takes on more of the form of provider–customer. Once again, APIs play the crucial role of defining market boundaries.

Realizing the Service Marketplace

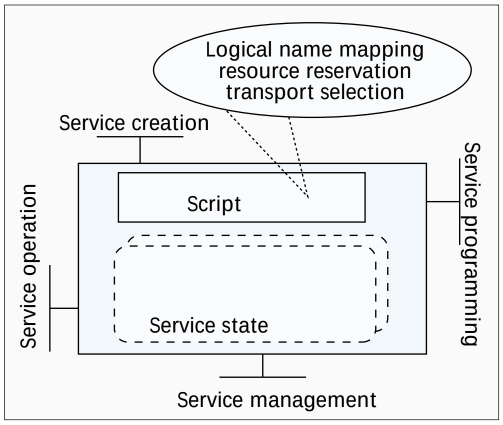
The layered market model outlined in the previous section is merely an economic model reflecting the author's vision of the future telecommunications service industry. In the remaining sections of this article, we focus on designing and realizing a novel service architecture in the technical sense which closely reflects the philosophy of the layered market model. The architecture we describe is targeted towards multimedia services as opposed to the economics of the general market model just presented. We begin by briefly describing the Extended Reference Model (XRM) for multimedia networks and its decomposition into three submodels [16].

The Extended Reference Model

The XRM models the communications architecture of networking and multimedia computing platforms. It consists of three components called, the *Broadband Network* (the R-model), the *Multimedia Network* (the G-model), and the *Services and Applications Network* (the B-model, see Fig. 2). The broadband network is defined as the physical network that consists of switching and communication equipment and multimedia end-devices.

Upon this physical infrastructure, resides the multimedia network whose primary function is to provide the middleware support needed to realize services with end-to-end QoS guarantees over the physical media-unaware network. This is achieved by deriving from the broadband network a set of QoS abstractions. The latter jointly define the resource management and control space (see also Fig. 6).

The process of service creation calls for resource reservation and distributed state manipulation algorithms. From this perspective, the multimedia network provides a *programming model* that allows service behavior to be specified and executed. Service abstractions represent the states of a service created using algorithms native to the multimedia network. These abstractions are used by the services and applications network for managing and creating new services through dynamic composition and binding.



■ Figure 3. *Interfaces for a G-model multimedia service.*

The Power of APIs in the XRM

As was mentioned previously, the functionality and indeed the level of sophistication of a service architecture is chiefly characterized by the APIs it offers. In the XRM, two types of APIs are available for building services. These are represented by QoS and service abstractions that lie at the interfaces between the R- and G- (also denoted R||G)models, and G- and B-models (G||B), respectively.

The primary power of these APIs is the tremendous flexibility available for service providers and users alike to mold the structure of the network in a way that reflects economic policies and business practices. By this we mean that these various levels of APIs allow different parties or stakeholders to influence the partitioning of resources to carve out natural market niches or even create wholly new markets. In other words, unlike the service architectures of the Telephone Network or the Internet which tend to fall along the lines of "all or nothing," we envision a future service marketplace to be rich in choices, variations, and sophistication.

R||G Interface APIs: QoS Abstractions

R||G interface APIs abstract the states of local multimedia resources (both logical and physical) in the broadband network. These resources represent the devices, switches, links, processors, and their respective capacities. Resources representing name spaces (e.g., VP/VC Identifiers) are also modeled. In the XRM, R||G interface APIs are implemented as a collection of distributed autonomous software entities with open interfaces. The open interfaces allow the states of these resources to be monitored, controlled, and managed remotely. More importantly, they allow these resources to be treated as independent pluggable components and services to be built by cleverly interconnecting combinations of them.

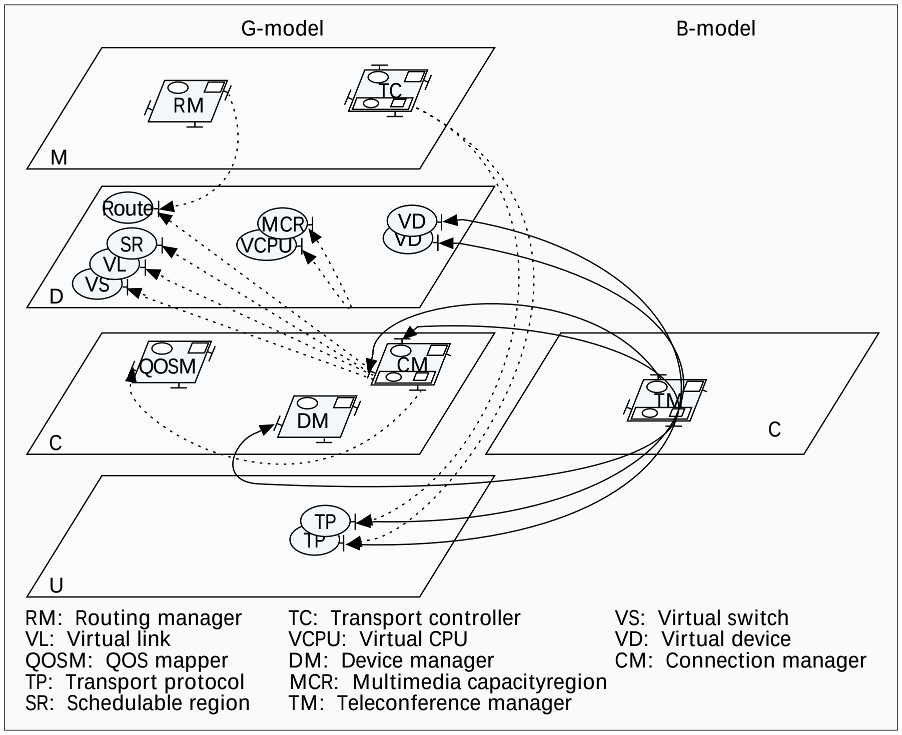
From the perspective of the G-model, the R||G QoS abstractions appear as a collection of interfaces called the Binding Interface Base (BIB) [2]. Because the APIs in the BIB are seen as basic building blocks of a multimedia network, they are key to several important new initiatives including open signaling (see OPENSIG [17]) and multimedia integration frameworks (such as DMIF within MPEG-4: http://drogo.cselt.it/mpeg/mpeg.htm).

We have recently announced completion of the draft specification of the BIB [18] at the OPENSIG workshop in Cambridge. The draft document and associated IDL templates have been made available to the community for comments and improvement. Also within ISO MPEG-4, a proposal for using the BIB as an interface framework for the "DMIF Network and Media Dependent Parts" has been put forward by Nortel [19, 20]. It is expected that this work along with the rest of the MPEG-4 proposals will reach international standard status by 1999.

G||B Interface APIs: Service Abstractions

In contrast to the R||G interface APIs, the G-model APIs provide access to the basic resource allocation and management services (realized as algorithms of the multimedia network).

These algorithms operate on the BIB with the goal of implementing a set of rudimentary communication services that allow the creation of simple point-to-point connectivity with guaranteed service characteristics. Collectively, these services are termed the Broadband Kernel Services (BKS) and are likened to networking counterparts of low-level operating

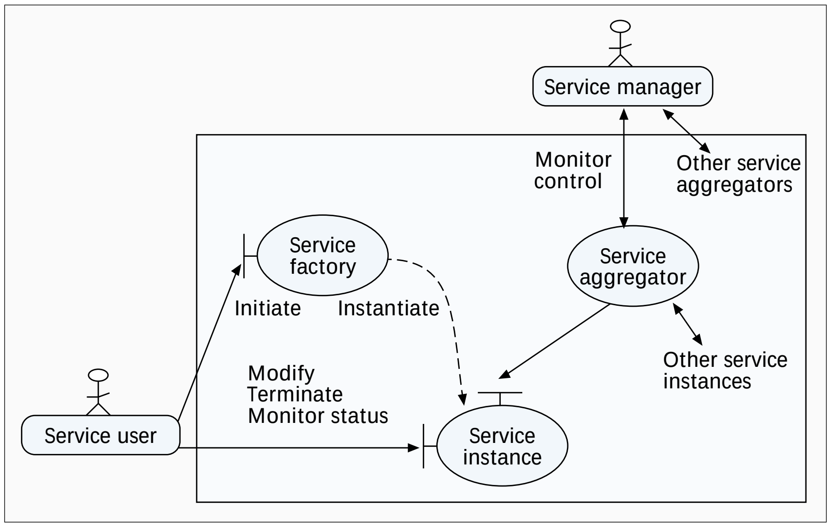


■ Figure 4. *Teleconference service invocations.*

system services (e.g., memory management, file system management, etc.). They
- Communication services such as connection management, routing, and admission control.
- Device management services.
- Transport level services such as transport monitoring and protocol stack management.
- QoS mapping services.

In a similar manner that BIB interfaces are assembled to create broadband kernel services, broadband kernel services can themselves be assembled together to compose even higher level services. An especially useful class of such services are network services such as:
- Virtual circuit services
- Virtual path services
- Virtual network services
- Multicast services

These services allow the construction of complex connectivity graphs in the network with associated transport and management facilities. The collection of network services at the G||B interface is known as the Broadband Network Services (BNS). Similar in concept to the BIB, the BNS defines a set of independent interworkable services that may be assembled to create yet higher consumer level services.



■ Figure 5. *Cooperating objects enabling service managment.*

The Service Creation Model

In accordance with the model described in the previous section, network services are obtained by invoking distributed algorithms in the space created by the BIB objects. For an object-oriented software implementation this requires modeling the service interfaces as well as providing a service creation model. Both will be described next.

Modeling Service Interfaces

High-level services (and conceptually all services) are composed of an algorithmic component and a data component. The algorithmic component expresses the execution logic of the service instance while the data portion is an abstraction of its state. In order for services to interact with each other, several types of service interfaces are defined. These interfaces reflect the roles that a service might play in the process of its execution. Typically these include *creation, operation, management, and programming.*

The service creation interface is akin a constructor of a class [21]. It is the primary entry point of execution or instantiation of a service and is called by the server once the service template has been completely downloaded. The service operation interface defines the operational functionality of the service and is usually the primary interface through which services interact. The programming interface allows manipulation of the service logic to be performed while a service is in execution. Finally the service management interface allows for monitoring of service states and manipulation of service parameters. This is illustrated in Fig. 3.

Recall from the previous section that the service creation process typically requires support for a number of middleware services. The bubble in Fig. 3 represents their invocation points embedded in the service script.

Although there might exist services with little or no state, the focus of our work is on those that require state keeping since these are typically more complex and harder to scale. Service states exist in two forms:
- Local states which are of purely local significance
- Distributed states which are states associated with other services and may be geographically dispersed

An example of a local state for a teleconference service might be the number of current users, while a distributed state might be the list of all switches through which a conference connection passes.

The Process of Service Creation: A Simple Example

The service creation process in broadband networks includes the creation of a service skeleton for an application, the mapping of the skeleton into the appropriate name and resource space, the association (or binding) to the application of a media transport protocol, the binding of the transport application to resources, creating a network service, and finally, the binding of the service management system to the network service, thereby creating a managed service [21].

As an example, consider a simple teleconferencing service which provides point-to-point video and audio connectivity services (see also Fig. 4). When a request for a conference session is received, a capability check must be performed at source and destination end-points to ensure that the appropriate multimedia devices exist and can support the requested media format. Next the appropriate end-to-end network resources are reserved so that a connection is established between source and destination nodes. This involves mapping the end points which may be specified in some logical names into physical nodes in the network as well as determining the route that a connection must take. Finally, transport selection and binding is performed so that both end-devices are associated with the correct network terminations.

Although the above model is realizable for simple services, the complexity of state management grows exponentially with the number of sub-services employed. As a result, complex services require additional support for state aggregation. This is covered in the next section.

Is the service creation methodology described above useful in the context of Internet applications? Take RSVP for example. Users are demanding the use of RSVP for controlling resources allocated to the various types of applications even if they are "RSVP dumb." For example, an user might be interested in assigning more resources to VLAN applications. This capability is hard to realize using end-system RSVP. However, it is easy to implement in our object-oriented context where a Connection Manager, implemented as a control object, can send upon request RSVP messages to routers supporting the VLAN application.

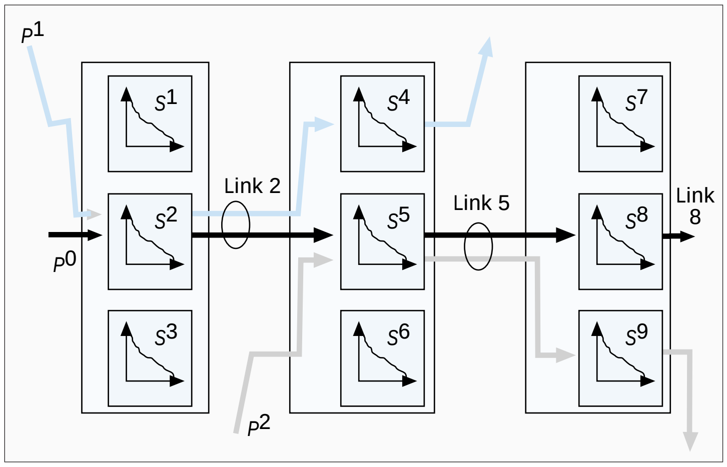


Figure 6. *Resource partitioning using schedulable regions.*

Managing Complex Services

Our approach (Fig. 5) for making a service manageable introduces a new type of controller in the service control system: a dynamic object that is created for each new session [22]. This object, which represents a service session (or service instance), makes itself accessible to the management system by registering with a management object, the service aggregator.

For each type of service there is a service factory which handles the requests for new service sessions and coordinates the service creation/instantiation process. For example, a service factory for unicast VCs receives requests for a new VC and contacts the necessary VC connection managers and routers in order to set up the new VC. The service factory abstracts the specific scheme used for the service creation process.

The service instance represents the status and capabilities of an existing service session. Every service instance exports two types of interfaces. One is accessed by the service user and represents the control capabilities (status, modify, terminate) a user has once the service session is created; the other is accessed by the management system and represents the management capabilities.

The service aggregator has two purposes. First, it provides a single point of access for the service manager to monitor and control the existing service instances of a particular service type. Second, it provides aggregated or abstracted views of the service instances and allows the manager to manipulate sets of service instances.

An important aspect of our model is that the designer of a service has several degrees of freedom when developing service management functionality. The choices include the selection of the state information and functionality of the service instances, the amount of information kept in the service aggregators, and the consistency requirements for the management data in the aggregators. All these factors influence the cost of managing the service in terms of design complexity during the development phase and in terms of processing and computational resources needed during the operational phase.

The Resource Allocation Model

The resource allocation model is based on game theoretic principles [23, 24]. By taking an economic (market-based and game-theoretic) approach in the engineering of multimedia networks, we seek solutions where the intelligence and decision-making are distributed and thus scalable, and the objective of a more efficient and fair utilization of shared resources results from the induced market dynamics. Thus, we are borrowing and adapting the tools of game theory from Economics and using them to solve problems of resource allocation (such as capacity allocation [25], virtual path bandwidth allocation [26], routing [27], and flow control [28]) in multiservice networks. By viewing a network as a collection of resources which users are selfishly competing for, this approach gives rise to efficient, decentralized algorithms, and leads to network architectures which provide explicit QoS guarantees.

The players in the network economy are software agents, rather than humans [29]. Agents acquire resources, such as bandwidth and buffer space, from the network on behalf of applications (video, voice, data transfer). Under appropriate rules of interaction, the collective actions of all the agents constitute a distributed intelligence, superior to that of any single controller. Thus the challenges are to analyze noncooperative behavior and algorithmic strategies, and to design the mechanisms (rules of the games) that will ensure the desired outcomes [30].

Resolving the contention for network resources among the various broadband network services is one of the key challenges of the XRM. In our context virtual circuits (VCs), virtual paths (VPs), virtual networks (VNs), and multicast (MC) represent the main broadband network services.

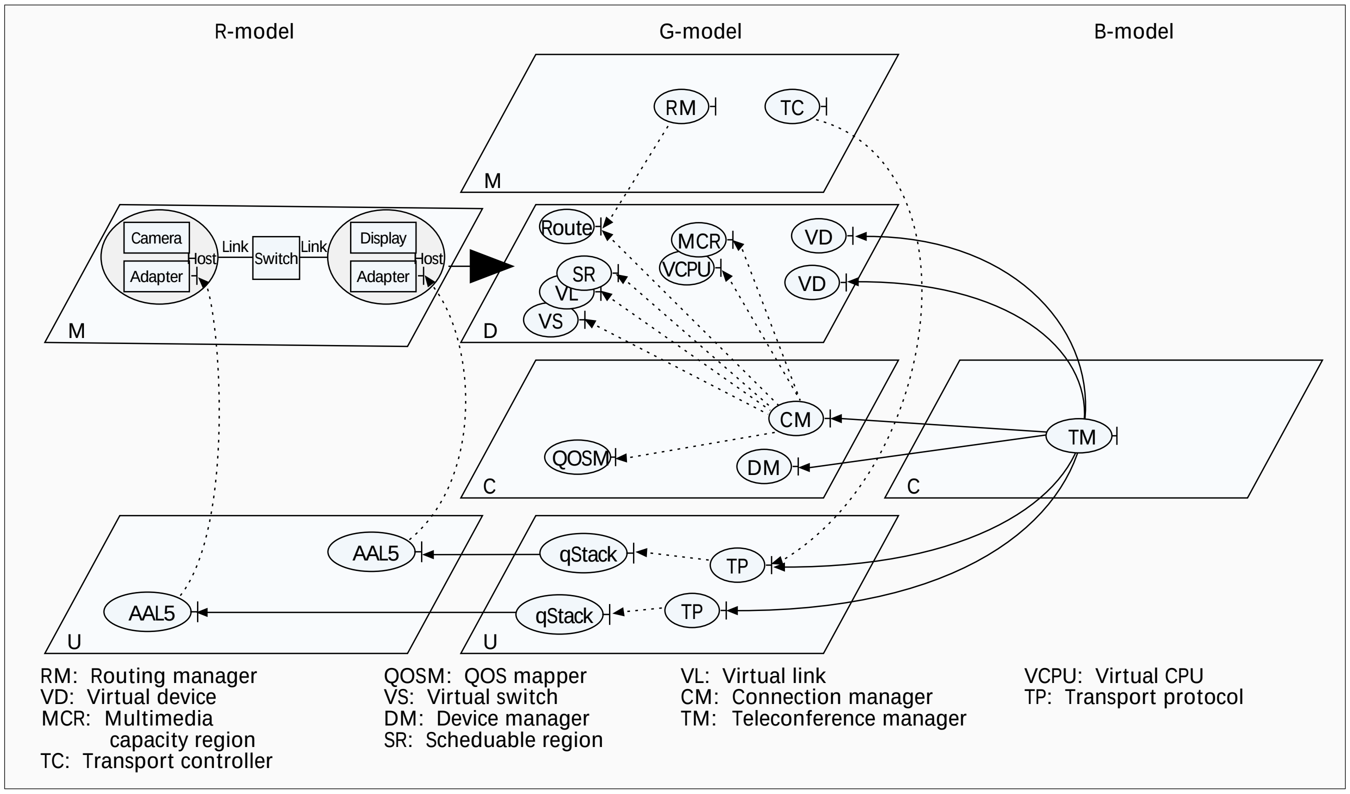
VCs, VPs, VNs, and MC can be modeled as capacitated graphs competing for inclusion into the network resource control and management space [31]. The latter is provided by the interconnected set of schedulable regions. Hence, the contention among the broadband network services has been reduced to a contention problem among capacitated graphs. We call this the *service graph contention problem*. In Figure 6 multiple decentralized controllers attempt to reserve VP resources such as bandwidth and buffer space [32]. The VPs in Figure 6 could be readily replaced with VCs, MCs, or VNs and the service contention problems among the latter similarly defined. In a more general setting all these services might compete for network resources at the same time and in various parts of the network.

How are these contention problems typically resolved? QoS support can be locally achieved via resource reservation algorithms. Recursive application of these algorithms along the capacitated graphs leads to a partitioning of network resources [31, 33]. Resources here refer to both name space, bandwidth, and buffer space. Thus resource partitioning algorithms become the key tool for efficiently guaranteeing QoS to broadband network services.

There are essentially two main categories of algorithms that can be designed for resolving the service graph contention problem. A competitive scheme among connection managers or a cooperative scheme that can be managed by a separate entity that is independent of the connection managers or switch controllers. See [34] for a competitive scheme in the context of VCI allocation. (A competitive scheme is distributed and works with local information. A cooperative scheme tends to be centralized and works with global information.)

Architecting the Multimedia Network

In the previous sections we described the XRM and the associated service creation and resource allocation model for realizing the service marketplace. We will now focus on architectural issues as well as QoS, performance, scaling, and implementation considerations arising in the design and implementation of an open programmable platform for service creation on ATM networks. This platform has been implemented at Columbia University as a broadband kernel called xbind [4]. Further information including documentation and software of our first prototypes is available on the Web [5].



■ Figure 7. *xbind system architecture.*

The xbind Broadband Kernel

In the fall of 1996, we completed implementation of a flexible open programmable networking platform for creating, deploying, and managing sophisticated next-generation multimedia services called xbind. xbind is conceptually based on the G–model of the XRM. The power of this conceptual framework comes from its seamless integration of elements from the domains of signaling (or connection management), transport, and management. This synergy allows traditionally difficult QoS networking issues to be addressed in an elegant and natural way.

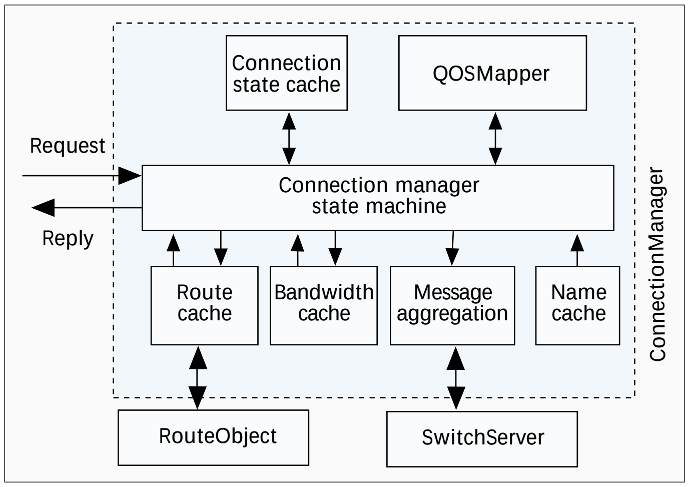
Functionally, xbind can be viewed as a broadband kernel for multimedia networks that guarantees QoS. xbind includes software components for implementing mechanisms for distributed network resource allocation, broadband signaling, realtime multivendor switch control, and multimedia transport and device management. It is based on the industry standard distributed object-oriented platform called CORBA and supports a simplified BIB, prototypes of all broadband kernel services, and a simple teleconferencing service with QoS renegotiation capabilities.

In xbind, all controllers are modeled as objects interacting through (remote or local) object invocations defined using the Interface Definition Language (IDL). Figure 7 gives the high-level system architecture of xbind. Four functional planes and the important objects on each plane are highlighted. The information transport, or G::U-plane, contains objects that handle transport activities — e.g., transport protocols and stacks. Objects on this plane interact with primitives of the ATM protocol stacks (e.g., AAL5) in the R::U-plane. Above the G::U-plane, the interactions among objects are on the signaling level. The Binding Interface Base (G::D) is a collection of CORBA interfaces that offers an abstract view of resources. These include naming resources like ATM Virtual Circuit Identifiers (VCIs) or physical resources like multimedia devices. Calls are made to these interfaces for creating low level services, i.e., broadband kernel services. Network services, using BKS, are built upon the BIB. The latter provide service abstractions and support to the B-model for building more complex multimedia services. Examples include routing, resource reservation, device management, and transport control.

xbind exploits the advantages offered by IP and ATM technologies without necessarily suffering their shortcomings. This is based on the observation that while typical interobject communication modes are mostly restricted to exchanges of short messages, the transport of multimedia streams involve typical call holding times several orders of magnitudes longer in duration. Furthermore, while interobject communication requires reliability even under adverse partial failure conditions, multimedia stream transport requires strict guarantees on delay bounds and losses. In this respect, it seems almost natural to use IP for interobject communication (which we term signaling) and ATM for media stream transport.

The xbind 2.0 (version 2.0) broadband kernel has been installed on a testbed of four ATM switches and interconnects the Columbia University distance learning program (the Columbia Video Network), to a wide area ATM testbed. In terms of platforms, xbind 2.0 has been ported to SunOS, Solaris, HP-UX, and Windows 95/NT operating systems and Fore ASX100/200, NEC Model 5, ATML Virata 1, and US Robotics TOTALcell 200 switches. The system also supports multimedia devices ranging from high-end workstation based JPEG video cards to entry-level real-time MPEG-1 encoders on the PC. The state of the ATM switches is mapped into the BIB using the GSMP and qGSMP protocols. The Connection Managers reside on the Sun Ultra 2 and HP 7000 processors. xbind 2.0, which is CORBA 2.0 compliant [35, 36], stands at roughly 30,000 lines of C++ and Java code. Access to the states of the switch hardware is via the GSMP and qGSMP protocols.

In conclusion, xbind's object-oriented signaling architecture is built on channels that are exclusively IP based, while the transport mechanisms are native to the underlying ATM network technology.



■ Figure 8. *Structure of the high performance connection manager.*

Mapping QoS Abstractions

In order to control the QoS at each contention point in the network a QoS abstraction on the object level is needed. Mapping of the states of the hardware into the software domain can be achieved with the QoS-extension of GSMP (the latter is a protocol proposed by Ipsilon Networks [37] and is now an Internet RFC) that we proposed and refer to as *qGSMP* [38]. The extension provides a number of key features, including the means of specifying QoS constraints, selecting scheduling and buffer management policies, and transfering schedulable regions [11, 13].

The extensions provided by qGSMP focus on controlling the output multiplexers and retrieving schedulable region estimates. They provide means for selecting scheduling, buffer management, and schedulable region estimation algorithms; setting traffic parameters and QoS constraints; and collecting QoS-related measurements. The general architecture of a switching environment using qGSMP is available on the Web (http://comet.ctr.columbia.edu/specs/specs.html).

The design of qGSMP follows the spirit of GSMP in the sense that it provides standard interfaces for development of switch-control software. Apart from semantic changes to two fields associated with the connection management messages, qGSMP does not alter the original GSMP messages. Moreover, the changes to these two fields are such that qGSMP is backward compatible with GSMP version 1.0. By providing generic descriptions for expressing QoS-semantics, the controllability and programmability of the switch control software is significantly enhanced, allowing possibly much better resource utilization.

Our implementation of qGSMP is supported by Advanced Telecommunications Modules Limited (ATML) switching technology. We plan to install the qGSMP interface on the switches provided by Washington University to the research community under NSF support. Finding efficient real-time algorithms for estimating the schedulable region represents a key research priority. Having access to the schedulable region will create an extraordinary opportunity for experimentation with resource partitioning algorithms that guarantee QoS. We have devised such algorithms based on game theoretic models in the past and believe that these can be readily implemented on our operational network.

Performance of the Service Delivery System

The service creation methodology described in the fifth section focused on the efficient realization of services in terms of the number of states and the flexibility in accessing them. The design trade-offs that our architectural model is enabling pertains to the location and the degree of cooperation among various entities that the service creation process is based upon. Service creation can be executed at the periphery of the network, as in the case of the Internet, or in the network itself, as is the case in the Telephone and ATM Networks. In the framework of our architecture there is complete freedom in locating the objects participating in the service creation process. In addition, there might be multiple providers offering services at various level of abstraction and thereby a parametrization between the “all or nothing” capabilities of the Internet and Telephone Network is made possible.

With the objective of supporting a large number of users, it is essential that the connection management system exhibits high performance. In [34], the elements of an approach for realizing a high performance connection manager with high *call throughput* and low *call setup delay* is presented. (Work on evaluating and improving on the performance of CORBA and Real-Time CORBA based distributed systems appears in [39].)

An efficient design of the connection manager has to take into account that the most expensive operations in a distributed environment are remote object invocations. In particular, the vast majority of the remote operations during connection setup have small arguments, remote calls contribute the bulk of the latency in call processing, and most computations are executed in the communication layer.

In order to design a high-performance connection management system, the following design criteria are recommended:

Parallelization of the Object Call Request Execution — design the system to run with a maximum amount of parallelization so that independent operations may proceed in parallel on idle CPUs.

Caching of Network States — minimize the number of remote procedure calls through aggressive caching of network states (network resources cache include name space and bandwidth).

Aggregate Access to Remote Objects — aggregate access requests to remote objects as much as possible.

A first version of the connection manager based on the above criteria has been implemented (Fig. 8). Running on a cluster of SUN Sparc 10s, initial results indicate that the system can support a throughput of about 100 calls/sec (or 360,000 calls/hour) with average latency of 200 ms. The intrinsic 16 ms call set up time is substantially shorter then the latency data published by the ATM Forum [40].

We believe that understanding the fundamentals of allocating the resources involved, in particular the distributed allocation of the VCI space, will allow us the tuning of the key parameters resulting in a further two to three times performance improvement. This level of performance will bring us in the range of the current STP processors.

Scaling Issues

The implementation of the service architecture on the xbind platform currently supports experimentation with small networks consisting of six to eight nodes. Scaling our real-time environment is fundamentally limited by costs, however. To experiment with the scaling properties of our architecture, we have realized, based on a parallel simulator, a high-performance emulation platform called TeleSoft. We intend to use this platform to study scaling aspects of the architecture and of the various telecommunication services.

The platform we have built over the last two years allows us to closely approximate the functional and dynamic behavior of network control systems on the call level [41, 42]. By providing support for real-time visualization and interactive emulation, it can be used to study telecommunications systems in various scenarios, such as different load patterns, network sizes, and management operations.

The emulated system and emulation support modules consist of a set of objects that communicate by exchanging messages, using functions provided by the simulation kernel. The emulated system module represents the prototype system under evaluation. Generally, each controller is implemented as a C++ object. Objects that interact with the parallel simulation kernel require minimal knowledge about the kernel — mainly how to send and receive messages. Therefore, the design of the emulated system follows the same rules as the design of controllers that run on a real network platform. The major difference is in how the interaction among controllers is realized. In the emulated system, interaction is performed by the simulation kernel. In a broadband network environment, the exchange of messages is provided by a signaling system.

Both the emulated system and the simulation kernel (also coded in C++) run on an SP2 parallel processor located at the Cornell Theory Center (CTC) in Ithaca, New York. The real-time visualization and interactive control module resides on an SGI Indigo2 workstation at Columbia University. It is written using Open Inventor, a 3D graphics tool kit based on Open GL, and runs as a UNIX process that communicates with the emulation support module via `UNIX System V` shared memory. The emulation support module is distributed on the two machines. These machines communicate through NYNET, an ATM network that connects several research laboratories in New York State. For more details as well as for download of the TeleSoft source code, the reader is directed to http://comet.ctr.columbia.edu/software.

Implementation Issues

Object-oriented methodologies and technologies are essential to us for developing service control and management systems. The basic elements of an object model, namely, concurrent objects interacting via messages, can be used to model controllers and their interactions in a distributed system. Such a model can be executed in a PDES (parallel discrete event simulation) environment (such as TeleSoft), as well as on a CORBA-based platform (such as xbind).

Our requirement is that the implementation design of the service control and management systems can be done in such a way that their code runs on both platforms. Also, the GUI enabling service management functionality by an operator must run on both platforms.

The servers or controllers represent perhaps the most complex implementation challenge in the model. The primary difficulty comes from the inherent flexibility a server must support — the ability to load any service script, instantiate, externalize, transport, and store its code and state and resolve all references to calls for broadband kernel services. The major implementation language used is Java because of the easy availability of its virtual machine and class loader facility. The API stubs to the broadband kernel services are implemented to make CORBA calls (or alternatively Java RMI calls) are themselves Java classes. Several Java based ORBs and interworking products provide this capability. Moreover, the Sunsoft JDK 1.1 release includes facilities for serializing [43] and externalizing Java object graphs. These facilities will be used to implement the basic facilities component in the servers. It is anticipated that the B-model services will be readily implementable in Java using the Java beans [10] component framework. This is because these services typically do not have to deal with hardware or I/O which are usually lacking in Java APIs.

Consequently, there is a wide range of possible designs for architecting the broadband kernel from light-weight, low-cost systems with minimal functionality to heavy, highcost systems with rich functionality. More precisely, we believe that a trade-off analysis must be made, balancing various factors in order to achieve the best combination between low cost, good scalability, and rich functionality. There are strong reasons to engineer the broadband kernel as a configurable system which can be customized at the time of service deployment. Moreover, it may be necessary to allow for dynamic reconfiguration of the system during run-time, in response to changing needs and the availability of resources.

Concluding Remarks

The major theme of this article is the modeling and realization of open programmable telecommunication networks. Specifically we described:

- A new architectural foundation for the creation, deployment, and management of future telecommunications services based on the paradigm of open programmable networking;
- Open Application Programming Interfaces (APIs) for service creation, QoS control, and the joint allocation of computing and communication resources;
- A service creation and resource allocation model;
- The need for investigating scaling issues through the emulation of complex service scenarios arising in large scale broadband networks;
- A methodology for evaluating the performance of the proposed service delivery system.

Due to space limitations, we have not discussed other key issues regarding the design and implementation of a service architecture for broadband networks. In particular, the extension of the service architecture to wireless environments (http:// comet.ctr.columbia.edu/wireless), problems of security, problems of software reliability, and, more generally, problems of verification and testing of the proposed architecture have not been mentioned at all. In addition, we have not touched upon research on active networks [44], a collection of strategies that allow network switches to perform customized computations on the messages flowing through them.

In conclusion, we believe that building programmable telecommunication networks is one of the key research challenges that faces the networking community as we move toward the new millennium. To address this challenge we have initiated a number of new projects and international forums (OPENSIG and OPENARCH, http:// comet.ctr.columbia.edu/ openarch) to promote the ideas which drive our research. We presented a number of research topics associated with service creation, QoS, performance, and scaling. Our work is backed by two software platforms currently in various phases of development. The xbind 1.0 platform is currently being used by a number of universities and industrial research laboratories worldwide. (See, among others, the list of participants in [17]).

Acknowledgments

The author would like to thank Koon-Seng Lim and the other members of the COMET Group, for extensive discussions on these and related issues. Thanks also are due to Jit Biswas, Fernando Cuervo, Ngoh Lek Heng, and Koon-Seng Lim who made valuable comments on an earlier version of the manuscript. The research presented here was supported by the Air Force Rome Laboratory, CSELT, HP, ISS, NEC, NTT, NYNEX, ORL, Sun Microsystems, and Toshiba.

References

[1] K. J. Willets and E. K. Adams, "Managing a Broadband Environment: You Can't Buck the Market," *IEEE Commun. Mag.*, vol. 34, no. 12, Dec. 1996, pp. 108–112.

[2] A. A. Lazar, S. Bhonsle, and K. S. Lim, "A Binding Architecture for Multimedia Networks," *J. of Parallel and Distributed Computing*, vol. 30, no. 2, Nov. 1995, pp. 204–216.

[3] Sun Microsystems Inc., "The Java Language Environment," White Paper, Oct. 1995.

[4] M. C. Chan *et al.*, "On Realizing a Broadband Kernel for Multimedia Networks," *Proc. of the Third COST 237 Workshop on Multimedia Telecom. and Applications*, Barcelona, Spain, November 25-27, 1996, pp. 56-72; available at http://comet.ctr.columbia.edu/publications/conference.html.

[5] Project xbind, http://comet.ctr.columbia.edu/xbind.

[6] CCITT Recommendation I.312/Q.1201, Principles of Intelligent Network Architecture, CCITT, April 1992.

[7] H. Berndt and R. Minerva, "Service Architecture version 2.0," TINA Baseline Document, TB_MDC.012_2.0_94, TINAC, March 1995.

[8] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview," IETF RFC 1633, June 1994.

[9] CCITT Recommendation Q.700, Introduction to CCITT Signaling System No. 7, Fascicle V1.7, Nov. 1988.

[10] Sun Microsystems Inc., JavaBeans 1.0 API Specification, Dec. 1996, available at http://splash.javasoft.com/beans/beans.100A.ps

[11] J. M. Hyman, A. A. Lazar, and G. Pacifici, "Real-Time Scheduling with Quality of Service Constraints," *IEEE JSAC*, vol. 9, Sept. 1991, pp. 1052–1063.

[12] F. P. Kelly, "Notes on Effective Bandwidths," *Stochastic Networks: Theory and Applications*, Kelly, F. P., Zachary, S. and Ziedins, I.B., Eds., Oxford University Press, 1996, pp. 141–168.

[13] J. M. Hyman, A. A. Lazar and G. Pacifici, "A Separation Principle between Scheduling and Admission Control for Broadband Switching," *IEEE JSAC*, vol. 11, May 1993, pp. 605–616.

[14] L. Zhang *et al.*, "RSVP: A New Resource ReSerVation Protocol," *IEEE Network*, vol. 7, no. 6, Nov./Dec. 1993.

[15] TINAC, Service Architecture Version 2.0, Document No. TB_MDC.012_2.0_94, March 1995.

[16] A. A. Lazar, K. S. Lim, and F. Marconcini, "Realizing a Foundation for Programmability of ATM Networks with the Binding Architecture," *IEEE JSAC*, Special Issue on Distributed Multimedia Systems, vol. 14, no. 7, Sept. 1996.

[17] OPENSIG, http://comet.ctr.columbia.edu/opensig

[18] C. M. Adam *et al.*, "The Binding Interface Base Specification Revision 2.0," *OPENSIG Workshop on Open Signaling for ATM, Internet and Mobile Networks*, University of Cambridge, Cambridge, UK, April 17–18, 1997; also CTR Technical Report #475-97-09, Center for Telecommunications Research, Columbia University, New York, April 16, 1997, available at http://comet.ctr.columbia.edu/publications/standard_contributions/BIB.html.

[19] V. Balabanian, "Binding Interface Base Operations," MPEG97/M2015, ISO/IEC JTC1/SC29/WG11.

[20] V. Balabanian and F. Cuervo, "Proposal of an Interface Framework for Control of Bindings between DMIF Network and Media Dependent Parts," MPEG97/ M2014, ISO/IEC JTC1/SC29/WG11.

[21] A. A. Lazar, and K.-S. Lim, "Programmability and Service Creation for Multimedia Networks," *Workshop on Multimedia and Collaborative Environments, Fifth IEEE Int'l Symp. on High Performance Distributed Computing (HPDC-5)*, Syracuse, NY, Aug. 6–9, 1996, pp. 217–223; available at http://comet.ctr.columbia.edu/publications/conference.html.

[22] C. Aurrecoechea, A. A. Lazar, and R. Stadler, "Towards Building Manageable Multimedia Network Services," *Proc. IFIP/IEEE Int'l Conf. on Management of Multimedia Networks and Services*, Montreal, Canada, July 8-10, 1997; available at http://comet.ctr.columbia.edu/publications/conference.html.

[23] Y. A. Korilis, A.A. Lazar, and A. Orda, "Architecting Noncooperative Networks," *IEEE JSAC*, Vol. 13, No. 7, Sept. 1995, pp. 1241-1251.

[24] Resource Allocation and Networing Games, http://comet.ctr.columbia.edu/networking_games.

[25] Y. A. Korilis, A.A. Lazar, and A. Orda, "Capacity Allocation under Non-Cooperative Routing," *IEEE Trans. on Automatic Control*, vol. 42, no. 3, March 1997, pp. 309-325.

[26] A.A. Lazar, A. Orda, and D. E. Pendarakis, "Virtual Path Bandwidth Allocation in Multi-User Networks," *IEEE Trans. on Networking*, 1998, to appear. Available at http://comet.ctr.columbia.edu/publications/journals.html.

[27] Y. A. Korilis, A.A. Lazar, and A. Orda, "Achieving Network Optima Using Stackelberg Routing Strategies," *IEEE Trans. on Networking*, vol. 5, no. 1, Feb. 1997, pp. 161-173.

[28] Y. A. Korilis and A.A. Lazar, "On the Existence of Equilibria for Noncooperative Flow Control," *J. of the Association for Computing Machinery*, vol. 42, no. 3, May 1995, pp. 584-613.

[29] N.G. Aneroussis, and A.A. Lazar, "A Framework for Pricing Virtual Circuit and Virtual Path Services in ATM Networks," ITC97, Washington, DC, June 23–27, 1997. Available at http://comet.ctr.columbia.edu/publications/conference.html.

[30] A.A. Lazar and N. Semret, "Auctions for Network Resource Sharing," CTR Technical Report # 468-97-02, Center for Telecommunications Research, Columbia University, Feb. 1997. Available at http://comet.ctr.columbia.edu/publications/techreports.html.

[31] J. M. Hyman, A. A. Lazar, and G. Pacifici, "VC, VP and VN Resource Assignment Strategies for Broadband Networks," *Proc. of the 4th International Workshop on Network and Operating System Support for Digital Audio and Video*, D. Shepherd, G. Blair, G. Coulson, N. Davies and F. Garcia Eds., Lecture Notes in Computer Science, vol. 846, Springer-Verlag, 1994; available at http://comet.ctr.columbia.edu/publications/conference.html.

[32] N.G. Aneroussis, and A. A. Lazar, "Virtual Path Control for ATM Networks with Call Level Quality of Service Guarantees," *IEEE Trans. on Networking,* 1998, to apear. Available at http://comet.ctr.columbia.edu/publications/journals.html.

[33] S. Borst, and M. Debasis, "Virtual Partitioning for Resource Sharing by State-Dependent Priorities: Analysis, Approximations, and Performance for Heterogeneous Traffic," *Proc. 15th International Teletraffic Congress*, Washington, DC, June 23–27, 1997.

[34] M. C. Chan and A. A. Lazar, "Connection Services on the xbind Broadband Kernel," *OPENSIG Workshop on Open Signaling for ATM, Internet and Mobile Networks*, University of Cambridge, Cambridge, UK, April 17–18, 1997; available at http://comet.ctr.columbia.edu/publications/conference.html.

[35] Object Management Group (OMG), The Common Object Request Broker: Architecture and Specification, Rev. 1.2, Dec. 1993.

[36] OMG, The Common Object Request Broker: Architecture and Specification, Revision 1.1, OMG Document No. 91.12.1.

[37] P. Newman *et al.*, "General Switch Management Protocol Specification," Ipsilon Networks, Inc., Palo Alto, CA, March 1996.

[38] C. M. Adam, A. A. Lazar, and M. Nandikesan, "QOS Extension to GSMP," *OPENSIG Workshop on Open Signaling for ATM, Internet and Mobile Networks, University of Cambridge, Cambridge*, UK, April 17–18, 1997; CTR Technical Report #471-97-05, Center for Telecommunications Research, Columbia University, New York, April 16, 1997, available at http:// comet.ctr.columbia.edu/xbind/qGSMP.

[39] D. Schmidt *et al.*, "A High-Performance End System Architecture for Real-Time CORBA," *IEEE Commun. Mag.*, vol. 35, no. 2, Feb. 1997, pp. 72–78.

[40] A. Battou, "Connection Establishment Latency: Measured Results," ATM Forum Document, ATM_Forum/96-1472, Oct. 1996.

[41] M. C. Chan, G. Pacifici, and R. Stadler, "A Platform for Real-Time Visualization and Interactive Simulation of Large Multimedia Networks," *Proc. Fourth IEEE International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS 96)*, Honolulu, Hawaii, April 1996.

[42] M.C. Chan, G. Pacifici, and R. Stadler, "Real-Time Emulation and Visualization of Large Multimedia Networks," *Proc. of the ACM Multimedia, Demonstrations Program*, San Francisco, CA, Nov. 1995.

[43] Sun Microsystems Inc., Java Object Serialization Specification Release 1.2, Dec. 1996, available at http://chatsubo.javasoft.com/current/doc/serial-spec/serial-spec.ps

[44] D. L. Tennenhouse *et al.*, "A Survey of Active Network Research," *IEEE Commun. Mag.*, vol. 35, no.1, Jan. 1997, pp. 80–86.

Biography

AUREL A. LAZAR [F'93] (aurel@comet.columbia.edu) (http://comet.ctr.columbia.edu/~aurel/) is a professor of electrical engineering at Columbia University. His research interests span both theoretical and experimental studies of telecommunication networks and multimedia systems. The theoretical research he conducted during the 1980s pertains to the modeling, analysis and control of broadband networks. He was the chief architect of two experimental networks, generically called MAGNET. In the early 1990s his research efforts shifted to the foundations of the control, management and telemedia architecture of future multimedia networks. His involvement with gigabit networking research led to the first fully operational service management system on ATM based broadband networks. His management and control research pioneered the application of virtual reality to the management of ATM-based broadband networks. His current research in broadband networking with quality of service guarantees focuses on modeling of video streams and analyzing their multiplexing behavior, with emphasis on multiple time scales and subexponentiality. He was instrumental in establishing the OPENSIG (http://comet.ctr.columbia.edu/opensig) international working group with the goal of exploring network programability and next generation signaling technology.