Making Multimedia Networks Programmable

Andrew T. Campbell
Department of Electrical Engineering and
Center for Telecommunications Research
Columbia University, New York, NY 10027-6699
http://www.ctr.columbia.edu/~campbell/

1. Introduction

Broad access to open gigabit ATM technology proposed by the Gigabit Network Technology Distribution Program at Washington University will play an important role in enabling the emergence of new broadband networking technologies and paradigms. We consider a major thrust of the proposed program [1] to be the development of an *open programming environment* that facilitates the easy creation of distributed multimedia services on top of the proposed Gigabit ATM Network Kits [1]. By open, we mean that the proposed platform must support a set of Application Programming Interfaces (APIs) for resource control and management which third party service providers can use when developing new services. By programmable, we mean that these APIs should be 'high-level' enough to allow the service specification and creation process to be carried out via a high-level programming language utilising distributed systems technology.

Since the Fall of 1994 we have been experimenting with a programmable platform called **xbind** [2], which provides a midddleware toolkit for the creation, deployment and management of multimedia services on ATM networks with end-to-end QOS guarantees. The principle aim of the **xbind**, which is based on a Binding Architecture [3], is to make multimedia networking environments programmable. In this extended abstract we present the **xbind** foundations for programmability.

2. Network Programmability

The Binding Architecture (as illustrated in Figure 1) consists of an organized collection of interfaces, called the Binding Interface Base [3] (BIB) and a set of algorithms that run on top of these. BIB interfaces provide an open and uniform access to abstractions that model the 'local' states of networking resources. Binding algorithms play a key role in the service creation process through the process of interconnecting (binding) networking resources. QOS is explicitly modeled in the architecture via a set of abstractions that characterize the multiplexing capacity of networking and multimedia resources under QOS requirements. These QOS abstractions allow admission control tests to be executed in real time during the resource reservation process.

xbind achieves interoperability between the different resource control and management algorithms of a multimedia network by providing open access to the BIB. By adopting the Common Object Request Broker Architecture (CORBA), **xbind** achieves interoperability in terms of data representation across diverse end-systems and ATM switch architectures. Using CORBA for **xbind** signalling allows CORBA objects such as the virtualSwitch (see section 3.1) to reside directly on ATM switches and end-systems. By modeling end-system and network resources explicitly and representing them as CORBA objects, we provide a seamless distributed systems environment for end-to-end programmability of services with QOS guarantees.

3. Binding Architecture

The Binding Architecture is comprised of a number of key architectural elements: the BIB, a

switch control API, a set of broadband kernel services, and a model for developing high level broadband services from these elements.

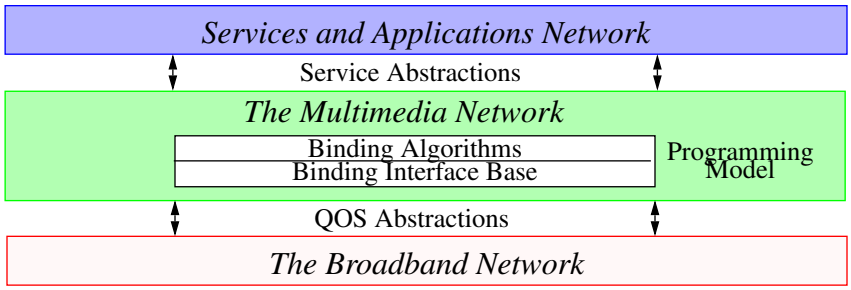

Fig. 1. Binding Architecture

3.1 Binding Interface Base

The Binding Architecture specifies all interfaces defined in the BIB in CORBA's Interface Definition Language (IDL). The IDL definition of the generic BIB factory interface called the BIBFactory allows the creation, destruction of an interface and querying of the list of interfaces currently instantiated by a server. All other factory interfaces are derived from this interface and customized to create specialized interfaces specific to the resource concerned. For example the VirtualSwitch controls the manner in which VPI/VCI pairs are allocated and deallocated. The VirtualSwitch interface provides access to the abstraction of the valid VPI/VCI pairs for both an ATM switch as well as an ATM adapter card. Obviously the two have substantial hardware differences: in the case of the switch the methods of the interface are mapped to the functionality of the hardware. In the case of the adapter, however, the fabric component is absent. We expect that in the future the internal bus of the work stations or PCs will have a design very close to an ATM switch and thus VCs will terminate not onto an adapter but directly to the target or source device. The primary methods of the VirtualSwitch interface are for requesting for free VPI/VCI pairs as well as for specifying the mapping between incoming and outgoing VPI/VCI pairs in a switch fabric.

3.2 Switch Control API

The implementation of the hardware specializations of the VirtualSwitch for all the platforms [2] required considerable development effort. Our goal of having access to low level control APIs that directly operate on the routing tables of various ATM switches is currently difficult to achieve since these are not open to third parties. However, through direct collaboration with the switch manufactures we were able to implement the VirtualSwitch on an individual basis for Fore Systems ASX-100 and ASX-200 switches, and NEC Model 5 switch. In the near future we will add the ATML Virata and Scorpio switches to the **xbind** testbed.

In order to speed up the implementation process, we proposed a set of APIs, called the Switch Control Application Programming Interface (SCAPI) [2], for the general control and management of ATM switches. The same motivation (albeit through a different approach) has also prompted Ipsilon Networks to propose a wired protocol called the General Switch Management Protocol (GSMP) with similar functionality to allow remote switch management. Main differences between the two proposals lie in the area of QOS control and their intended use. In the SCAPI proposal, QOS issues play a central role. In the current version of GSMP, however, QOS issues have not been explicitly addressed. Moreover while GSMP is meant to allow remote access to switch con-

trol functionality, SCAPI is a programming interface for developing high level control software on a switch. In the Binding Architecture, this is represented by the software that implements the VirtualSwitch interface.

3.3 Broadband Kernel Services

The Broadband Kernel Services refer to low level "operating system" like functionality provided by the middleware for deriving higher level multimedia services. These include services for connection management, routing, device management and transport. The Connection management service provides connection setup and tear down functionality between a number of network endpoints. The routing service provides a route location functionality for traversing any two network endpoints. Together, they implement the basic connectivity services in the network component of the system and can be used to built Switched Virtual Circuits (SVCs). On the host end, a device management service provides the functionality of tracking and managing the numerous multimedia devices spread throughout the system. Ensuring format compatibility and presentation translation are also aspects of this service. Finally the transport management service defines the functionality required for sustaining transport streams in the network. These include functionality for rate control and bandwidth renegotiation.

3.4 Broadband Services

Broadband Services are offered by servers which support the creation and maintenance of their state. When a request for a service is made to a server, a service instance is instantiated or created. Each service instance is composed of an algorithmic part and a data part. The algorithmic part expresses the logic of the service instance while the data portion is an abstraction of its state. Individual service instances can be customized by modifying the logic of its algorithmic part. Similarly control can be effected on an executing service instance by the modification of its states. Servers also have an algorithmic component which specifies how service instances are created, deployed and managed and a data part which models the state of the server and its policies. Typical servers expose 4 interfaces [2]. These are the Service Factory interface, the Service Programming interface, the Service Control interface and the Service Management interface.

4. Summary

Access to Gigabit ATM Network Kits [1] will enable new models for network signalling, control and management to be developed. Based on our experience with the **xbind** approach we consider a major thrust of a new gigabit program to be the development an open programming environment that facilitates the easy creation and management of distributed multimedia services. It is our conjecture that ATM networks should be made to go faster and become more programmable at the same time.

5. References

[1] Gigabit ATM Network Kits -- Research Technology Distribution: http://www.arl.wustl.edu/~jst/gigatech/ResDist.html

[2] Project **xbind:** http://www.ctr.columbia.edu/comet/xbind/xbind.html.

[3] Lazar, A.A., Bhonsle, S. and Lim, K.S., "A Binding Architecture for Multimedia Networks", *IEEE Journal of Parallel and Distributed Computing*, Vol. 30, Number 2, November 1995, pp. 204-216. Also in the Proceedings of the Multimedia Transport and Teleservices, Vienna, Austria, November 14-15, 1994.