MMCN97

Active multicast service architecture for user customized multimedia data transmission over ATM networks

Hongyi LI[a], Hung Keng PUNG[†b], Lek Heng NGOH[b]

[a]Multimedia Systems Research Lab.,
Dept. of Information Systems and Computer Science
[b]Institute of System Sciences
National University of Singapore
Lower Kent Ridge Road
Singapore 119260

ABSTRACT

This paper proposes a new ATM multicast scheme within the framework of an Active Multicast Service Architecture (AMSA) that supports user customization of multicast service. The customization is achieved by injecting and invoking user specific programs to shared network resources such as routers and switches. The multicast scheme also allows dynamic joining and (or) leaving of members of different capabilities and quality of service requirements. A prototype of the multicast scheme was developed to verify the major functions of the AMSA and to demonstrate the feasibility of customizing network services within the active service architectural framework . The prototype supports QOS negotiation, non-uniform links QOS, resource reservation and filter program injection. The development platform used consists of virtual ATM switches (implemented as light-weight threads in four workstations for simulating real ATM switches) inter-connected via a real FORE ATM switch. The data links between these virtual switches are AAL-5 virtual circuits that enable actual QOS reservation. The signaling of the virtual switches is implemented via CORBA links connecting the CORBA daemons residing in the virtual switches. Experimental results show that the prototype of ATM multicast scheme is efficient and flexible in satisfying the requirements of heterogeneous users.

Keywords: ATM Multicast, Active Network, Multimedia Communication, CORBA, Network Simulation.

1. INTRODUCTION

The current standard of ATM signaling[4] only allows source initiated multicast connection setup and tear-down which is relatively restrictive in comparison with multicast scheme supporting receivers initiated joining and leaving. The reservation of network resources for multicast virtual channels must be performed by the source [5,12] and all the receivers must have the same Quality of Service (QoS) requirements, including the rate of data to be received. This rudimentary multicast support cannot satisfy the heterogeneous receivers with various capabilities and preferences. Furthermore, individual user has no mean of customizing the multicast services to adapt to its own environment and preference in current ATM network.

We propose here an an Active Multicast Service Architecture over ATM network that supports user customization of multicast applications by using active technologies. It enables users to deploy multicast services rapidly by selecting basic network service components, injecting customized programs, and finally tailoring them into a new type of services to suit their specific requirements. This approach is believed will accelerate the deployment of new network services as they are less prone for delay by vendor consensus and standardization activities.

The AMSA extends the active network concept of Tennenhouse of MIT by providing two levels of services for users to customize their views. In the first level, users may customize the applications by choosing certain standard services that are provided by the AMSA. For instance, users can select preferred routing services by emphasizing on various data transmission requirements, such as real-time, reliability, or cost effectiveness. In the second level, AMSA proposes the

---

[†] Correspondence: Email: punghk@iscs.nus.sg; WWW:http://www.iscs.nus.sg/~mrl; Tel: 65-7722808; Fax: 65-7794580.

injection and invocation of user specified programs in the switches for computation or processing of user data flowing through them. This requires the availability and support of programmable ATM switches with software control. Generally, user specified programs consist of simple instructions, such as data filtering, data compression, encryption algorithms and etc. AMSA utilizes a broker agency - the Multicast Broker Agency (MBA) in this case - to manage and control the dynamic process of user customization of a multicast service. The MBA contains several agents which are responsible for the monitoring of ATM link states, multicast routing, group management and connection management.

In Tennenhouse's active network, user data packets are encapsulated with user program fragments. These capsules travel to ' active nodes' where the capsules are interpreted[15]. Since we use a broker agency (or server) to facilitate the dynamic user customization process, our approach is different in the ways of providing services for invoking user specified programs in the network nodes. AMSA encapsulates only the program handler, whist user data and the user specified program fragments are invoked from the switches by the MBA. Our approach has several advantages. Firstly, it reduces the amount of data traveling along the network links because program fragments need not be encapsulated together with data packets. Secondly, the multicast routing is also performed in the MBA. This enables the user specified programs be invoked in the strategic points in the network. Moreover, a centralized user program injection mechanism makes it easier for applying security policies to the user customization processes.

Research on programmable ATM switch for supporting concepts similar to the active network concept is emerging. For instances, the X-Bind Model proposed by Lazar at Columbia university[6] and the SwitchWare approach proposed by University of Pennsylvania and Bellcore[18]. In the X-Bind concept, network components are modeled as objects in the object-oriented distributed programming environment and the objects can be bound for supporting specific applications. Furthermore, several companies have announced plans to develop so-called third layer switches to support mechanisms such as filter on high level attribute[3]. All these works give the support to our proposed architecture.

The idea of active network is not new. In the traditional computer network arena, there are several ad hoc approaches that allow user-driven computation at network routers or switches such as data filtering, firewalls, web proxies, etc. RSVP is a resource reservation protocol for supporting multicast applications over heterogeneous platforms. In RSVP, the filters are assigned in the various intermediate nodes during a multicast session to facilitate the selective data transmission. However, RSVP cannot guarantee end-to-end QoS due to the separation of routing and resource reservation. ST-II is another multicast protocol that allows user to specify QoS requirements and guarantees end-to-end bandwidth and delay for all connections. ST-II operates by layering a virtual circuit service on top of IP that is not compatible with the datagram service of IP such that it effectively reduces the number of ST-II users. Both ST-II and RSVP do not address routing or make a simple attempt on routing. The ST-II builds a multicast tree from unicast routing table while RSVP relies on the underlying network for routing. The solutions of these ad hoc approaches are not neat as there does not exist a generic architecture in current networks to allow users to program the network for special applications. Although active technologies are being applied at end-to-end level in the network, they have not been leveraged and extended for use inside the network[15].

AMSA is a powerful and flexible architecture that supports various network features required by user customization processes. It allows receivers to select preferred routing strategy, to initialize resource reservation for end-to-end QoS guarantee, and to invoke user specified programs such as filtering, buffering, encryption and compression programs. These services are achieved by using a centralized multicast broker agency which is responsible for QoS negotiation, multicast routing, connection management and user program invocation. Furthermore, the architecture supports nodes joining and leaving to a multicast group dynamically. This architecture is a complex system that integrates many network functions and supports many user specific properties.

Currently, we use virtual switch to simulate the activities of a programmable ATM switch and we expect that the functions of a virtual switch can be supported by real switch fabricate in the near future. A prototype system for evaluating the proposed architecture has been developed in a simulation platform that is built on the distributed CORBA environment. The simulation platform consists of four workstations that are connected to a FORE ATM switch. In this platform, the functions of a network switch are modeled as a virtual switch that is a concurrent thread running on one of the workstations. The data links between virtual switches are emulated by real AAL 5 virtual circuits that enable actual QoS reservation. Signaling of the virtual switches is implemented by CORBA links that connect the CORBA daemons reside in the virtual switches.

The organization of this paper is as follows. In section 2, the architecture of the active multicast service is presented and the important components of the architecture are described. The concepts, functions, and algorithms in Multicast Broker Agency are explained in this section. Further studies on data filtering techniques, QoS guaranteed routing algorithms, user program invocation techniques and dynamic members joining and leaving algorithm are also described. In section 3, a prototype system that implements the functionality of AMSA is presented. The implementation of the simulation platform, the virtual switches, the structure of the simulation platform, the topology of the simulation network, and experiments are described. Finally, in section 4, the conclusion is given and future works are discussed.

2. ARCHITECTURE OF THE PROPOSED ACTIVE MULTICAST SERVICE

The proposed AMSA can be modeled as a product distribution process that involves the interactions among Producers, Consumers, Brokers and Switches. The model can be expressed as a quadruple $\{\mathcal{P}, \mathcal{C}, \mathcal{B}, \mathcal{S}\}$, where $\mathcal{P}$ is the set of producers that are the sources of the information conveyed in the multicast sessions; $\mathcal{C}$ is the set of consumers that are the sinks of the information conveyed from the producers; $\mathcal{B}$ is a set of brokers in the Multicast Broker Agency (MBA) which is responsible for negotiation and implementing a multicast session among the producers and consumers; $\mathcal{S}$ is the set of switches that transports and relays the data streams from producers to consumers. The relationships among these components are depicted in Figure 1. The functions of $\mathcal{P}$ and $\mathcal{C}$ are straight forward while the functions of $\mathcal{B}$ and $\mathcal{S}$ will be elaborated in following paragraphs. The MBA consists of four agents: the Monitoring Agent (MA), the Group Management Agent (GMA), the Routing Agent (RA) and the Connection Management Agent (CMA). In order to collect link state information from or send signals to switches, the MBA maintains a set of permanent bi-directional connections to all the switches where the incoming links are used by MA for monitoring purposes while the outgoing links are used by CMA for signaling. The MBA connects to its clients (i.e. hosts) through a set of transient point-to-point bi-directional links. These connections are established by the hosts that require the multicast services from MBA. The hosts can send queries to or receive replies from the MBA.

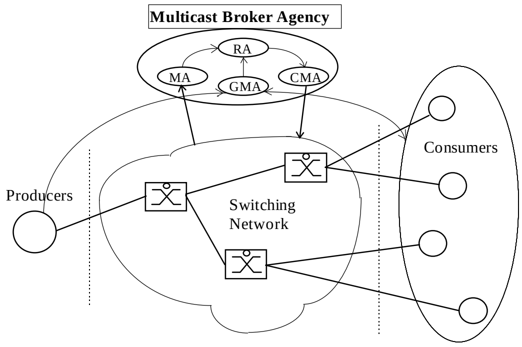

Figure 1. The active multicast service architecture

2.1 Network state monitoring

MA monitors the state of all the switches to collect link state information of the network. The task of monitoring can be implemented by using a topology state routing protocol in which switches send QoS and reachability information toward MA. Therefore, MA obtains knowledge about reachability and available traffic resources within the network. This is similar to the current link state routing protocols such as the OSPF. The current set of link state information collected by MA are: Maximum cell transfer delay (MCTD), Maximum cell loss ratio (MCLR) and Residual cell rate (RCR). The collected link state information can be used by RA for QoS guaranteed routing computation.

2.2 Multicast group management

GMA provides functions of group management and a channel for negotiation between producers and consumers. It manages the group membership status and member joining or leaving activities. GMA, analogous to the MARS in IP multicast over ATM[5], keeps an extended table of multicast group address for multicast address resolution. When a producer has packets for transmission, GMA is queried for the set of consumers currently constituting the group. Then GMA returns the addresses ATM.1, ATM.2, ..., ATM.n to the producer. In this case GMA is acting as a mediator between the producer and the consumers to facilitate their interactions (i.e. negotiation). The negotiation is a process of exchanging information between producers and consumers in order to reach an agreement for the permissible future data transmission. This agreement includes the terms of QoS parameters to be guaranteed during the data transmission and the user specified programs to be invoked in the switches.

2.2.1 QoS negotiation

It has been articulated by some researchers that ATM applications shall be able to request QoS parameters from the network, or in other words the parameters must be transferable to the applications through the upper-layer protocols[9,17]. Therefore a mapping mechanism is needed to transform the user QoS parameters into ATM QoS parameters. This involves a negotiation process which has two phases: an advertising phase and a selection phase. The producer advertises the multicast data description in the advertising phase. Assuming σ_0, is the multicast data source from the producer. A set of new data $\sigma_1, ..., \sigma_m$ can be derived from σ_0, by applying one or more times of the user programs. The set of all possible user programs associated with σ_0 can be expressed by $\Phi = \{\phi_0, \phi_1, ..., \phi_s\}$. Then we define a set of all possible data sources as $\Sigma = \{\sigma_0, \sigma_1, ..., \sigma_m\}$, where $\forall \sigma_i \in \Sigma$; $\exists\ \sigma_j \in \Sigma, \exists \phi_k \in \Phi$; such that $\sigma_i = \phi_k(\sigma_j) \in \Sigma$. Therefore, the contents of an advertisement message consist of a formula list for each σ_i where $\sigma_i = \phi^1 \bullet \phi^2 \bullet\bullet\bullet\ \phi^k(\sigma_j)$ and the $\phi^1, \phi^2, ..., \phi^k \in \Phi$. A small set of programs is identified that includes filtering, buffering, stream cipher, block cipher and some real-time compression programs. Users can select a sequence of these programs and invoke them from switches to process their data streams.

The advertisement messages are first sent to the GMA by the producer. The GMA then multicasts them to all the receivers of the specified group. In the selection phase, each receiver selects the desired data streams according to her preference and capability of the workstation and the network interface. The selected data stream for each receiver is passed to the routing agent for a routing computation.

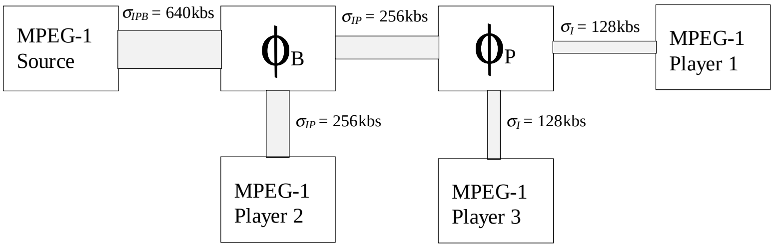

Figure 2. The effects of draping filters for MPEG video stream

Figure 2 shows a typical case where a combination of filters can be used on an MPEG-1 video stream (MPEG-1 is not chosen for its merits but for easy illustration of concepts). In this figure, we use two frame-filters, ϕ_P and ϕ_B, where ϕ_B drops B-frames and ϕ_P drops P-frame from a MPEG-1 video stream. These filters result in video streams of different quality as indicated by the thickness of the streams in the figure. In this example, the sender sends out σ_{IPB}; $\sigma_{IP} = \phi_B\ (\sigma_{IPB})$; $\sigma_I = \phi_P \bullet \phi_B(\sigma_{IPB})$ and the corresponding bandwidth for each stream as indicated in Figure 2.

2.3 Multicast routing agent

The Routing Agent computes a multicast tree that connects producer and consumers by using the link state information collected from MA and the group membership information and the negotiated agreement from GMA. It also offers routing options to the receivers that may emphasis different requirements on data transmission, such as reliability, real-time, or cost effectiveness of transmission paths. When RA computes the multicast tree, it selects the strategic switches to invoke user specified programs. The results of RA are a set of contracts for all switches involved in the computed multicast tree. The contracts are frames that describe the required services from a switch for a specified multicast session. A contract contains slots of session-id, input port, output ports and user programs associated with output ports, etc. The format of the contract is depicted in Figure 3.

```
Contract {
   sessionID:
   inPort:
   inPortQoS:
   outPort 1{
         outPortQoS:
         outPortMethod:
   }
   …
   outPort n {
         outPortQoS:
         outPortMethod:
   }
}
```

Figure 3. Structure of contract frame.

2.3.1 Multicast routing with QoS constraint

Traditional multicast routing algorithms were designed to support users with homogeneous and simple QoS requirements. With increasing demands for integrated services, the routing algorithms must support diversed, fine-grain and subjective QoS requirements. In a connection-oriented network such as ATM, the transferring of data between end-users is accomplished by network routing functions that select and allocate network resources along the acceptable paths. In our proposed scheme, routing is performed by a centralized RA. It is obvious that the merit of this type of routing partly depends on the accurate information of the network topology and the link state information. This is similar to the current link state routing protocol such as OSPF where all network nodes can obtain estimated current state of the entire network. We assume this information can be collected from switches by the MA. The current set of link state information collected by MA are: Maximum Cell Transfer Delay (MCTD); Maximum Cell Loss Ratio (MCLR); Residual Cell Rate (RCR).

The network can be modeled as a graph of nodes connected by node-to-node links. Each link has parameters Maximum cell transfer delay (MCTD), Maximum cell loss ratio (MCLR) and Residual cell rate (RCR). These values are known to the routing algorithm and we use t, e, and b to represent MCTD, MCLR and RCR in the distance function. A simplified distance function d_{ij} reflects transfer delay t_{ij}, cell loss rate e_{ij} and residual bandwidth b_{ij}. It is defined empirically for each direct link $l_{ij} : d_{ij}(t_{ij}, e_{ij}, b_{ij}) = \omega_1 t_{ij} + \omega_2 e_{ij} + \omega_3 (1/b_{ij})^{\alpha}$, where l_{ij} denotes the link from node i to node j. The exponential form of the residual bandwidth term compared with the transfer delay term's linear form shows the emphasis on bandwidth exhaustive as a much more important factor affecting the distance function. The ω_1, ω_2 and ω_3 are weights of the three factors that can be changed by the users in the negotiation phase. Increasing the weight of a factor may change the emphasis of the routing algorithm. For instance, if a receiver wishes to receive a higher reliable data stream, it can increase the value of ω_2 that leads the routing algorithm emphasizing more on the error free paths during the routing computation.

By applying the Dijkstra algorithm[1], the shortest path from source node to the receiver nodes can be computed. If the paths leading to different receivers travel along the same link, they need to be merged into a single path. After all the common paths are merged, a multicast tree can be established. In the merging process, QoS reservations for different receivers are to be aggregated to form a resource reserved multicast tree. As a result, only the essential (i.e. the minimum superset) common QoS is reserved on a link for a common path. A filter is assigned to each outgoing path that is splitted from the common link to facilitate the selective data transmission. These filters select the appropriate data sets for the specified receivers according to their QoS requests. Therefore, a logical multicast tree with various QoS reservations on different branches can be established. It should be noted that all the actions described in the above paragraphs, such as routing computation, resource reservation and filter placement, are performed in the memory of RA (i.e. we only have a logical multicast tree at this stage). Subsequently, the logical multicast tree will be converted into separate contracts for all the involved switches for interpretation and implementation.

2.3.2 Node joining / leaving a multicast tree

A new member can join an existing multicast group through a joining operation. In the operation, an unicast route is first computed from source to the joining node by the RA based on the QOS request. Then the joining algorithm will try to merge this new route to the existing routing tree. It first traces the new route from the joining node toward the source until it hits the first intermediate node of the existing multicast tree. This intermediate node is selected as the merging point for the remaining links to the source. The resource reservation for each link of the merged paths can be adjusted by a modify process which updates the resource reservation according to the new requirement. In the worst case, the attach point is the source. Figure 4 shows an example of node R4 joining an existing multicast tree in which the sender S is sending data stream to receivers R1, R2 and R3 via a switch network involving Sw1, Sw2, Sw3 and Sw4. The unicast route from source to receiver R4 is indicated by the dash lines. The attachment point for the joining node is in Sw2. The unicast route from S to R4 has a common link with the existing multicast tree at link between Sw1 and Sw2. This common link needs to be merged into the same path.

The leave operation removes the leaving node from all the multicast tree currently sending data to it. As in the joining algorithm, the node is disconnected with each multicast tree separately. For instance, if the receiver R4 wishes to leave the multicast tree as shown in Figure 4. The leaving algorithm first disconnects the R4 and then releases the reserved resources. The intermediate switch SW5 checks if there exists other nodes still attaching to itself in the multicast tree. It will remove itself from the multicast tree if there is no group member attached to it. Otherwise, the intermediate node frees the resources that were reserved for the leaving node. The resource reservation on link

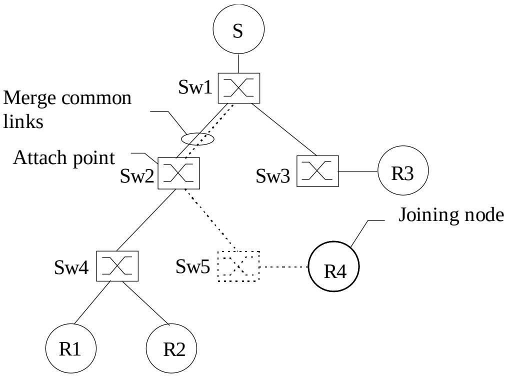

Figure 4. Dynamic member joining / leaving scheme

SW1-SW2 also need to be adjusted accordingly.

2.4 Connection management

Once a routing contract for each switch is worked out, the CMA is invoked to setup the actual multicast tree. This is done by sending the routing contracts to the designated switches via the links between MBA and the switches. A multicast channel in a switch should have one incoming VPI/VCI and multiple outgoing VPI/VCIs. The routing contract uses the inPort and the outPort to specify the switch to be connected to the incoming VPI/VCI and the switches to be connected to the outgoing VCI/VPIs respectively. When switches receive their routing contracts sent by the MBA, each of them can start to signal the upstream and downstream switches for establishing the corresponding VPI/VCI's. This signaling process is a parallel process in which each switch contacts its related upstream and downstream switches independently. In this way, the switches, producers and receivers can be bound into one multicast session rapidly. As depicted in Figure 5, the signaling at each switch has two phases namely *connect* and *accept*. For instance, Sw2 sends the connection signal with a QoS specification to all the downstream switches (i.e. Sw3, Sw4). The downstream switches will send their acceptation messages and actually setup the links if the required QoS can be satisfied. Then Sw2 extracts the user program fragment from the contract and associates it with the specified outgoing link. These user program fragments are stored as libraries that can be invoked dynamically to process outgoing packets.

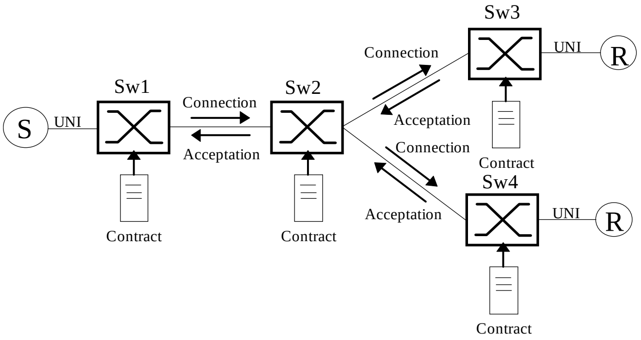

Figure 5. Signaling scheme

2.5 Interaction among agents

The four agents communicate with each other as depicted by the MBA part in Figure 1, where MA and GMA send link state information and negotiated multicast session agreement to RA. The RA works out the routing contracts and sends them to the CMA for setting up the multicast tree. In this model, the tasks of MBA are performed in a centralized fashion that benefits the multicast service management in following aspects: Firstly, the GMA provides uniform QoS negotiation channel that normally cannot implemented in the current ATM multicast supporting only the unidirectional virtual channels. Secondly, the centralized RA makes the computation of multicast tree easier to retrace, revise and avoid loops when computing on a centralized network map. Finally, the CMA reducing the number of required signalings for routing at each switch. Although a centralized approach also has several setbacks such as potential single point of failure, traffic congestion and other scalability problems, there are techniques to overcome these setbacks. For instance, a backup MBA can be used to protect the MBA service from single point failure. Since a MBA is used for managing a cluster of ATM switches (e.g. 64 switches), a high performance workstation is able to handle signaling traffics and routing computation for the switch clusters of reasonable size.

3. A PROTOTYPE OF ACTIVE MULTICAST SERVICE ARCHITECTURE

A prototype system to implement the proposed architecture has been developed in a simulation platform that is built on virtual switches. The platform consists of one ATM switch (i.e. a Fore switch) connected to several workstations. Each workstation runs a Orbix server to signal and control the virtual switches. A virtual switch is represented by a thread in a specified Orbix server that resides in one of the workstations. The Multicast Broker Agency is simulated as an Orbix client that performs the functions of MA, GMA, RA and CMA. In the simulation platform, the links connecting virtual switches are emulated by real ATM links that connect multiple workstations via the Fore ATM switch. The signals between MBA and producer, consumers and switches (i.e. virtual switches) travels along the Orbix links.

We use this prototype system to verify the major functions of the Active Multicast Service Architecture. Firstly, the prototype system allows the receivers to customize a MPEG-1 video multicast session to fit their specific requirements. For instance, a receiver can select high, medium, or low quality of video streams and then the MBA to invoke the corresponding filter programs into the virtual switches. These filter programs are attached to the specified outgoing links of a multicast channel in a virtual switch. A handler of filter program is encapsulated in the data packet sent by the producer. When a packet arrives at a switch, the handler encapsulated will be evaluated by the filtering program to decide whether to drop this packet or not. These filter programs are very simple such that virtual switches only evaluate the header of the capsules. In addition to selecting filters, the receivers can also select different routing strategies by varying the weight of each routing factor, such as delay, cost and error rate. The prototype system validates a new signaling scheme between virtual switches that enables the autonomy virtual switches to interpret routing contracts and to signal neighboring switches independently in accordance to the contract. This signaling scheme implements parallel setting up of multicast tree in an ATM network and simplifies the task of the central control mechanism.

3.1 Simulation platform

It is more meaningful to study a multicast routing scheme in a ATM network of reasonable size. Due to the relatively small number of ATM switches installed in our test-bed, we resort to simulation technique using virtual switches implemented as Orbix objects simulating the behavior of the real switches. The simulation platform is established in a distributed environment with four workstations connected to a FORE ATM switch as depicted in Figure 6. The reason of using four stations is based on the theory that any region in a map can be labeled by four colors that guarantees no adjacent regions having the same colors. Therefore, to simulate any network topology the virtual switches can be run in the four workstations such that any two virtual switches are guaranteed to be simulated at different workstations and be connected by a ATM VC over the FORE ATM switch. This connection strategy facilitates the emulation of the QoS reservation for each link in the multicast tree. As indicated in Fig 6, the link between node 1 and node 2 of the multicast tree is mapped to the VC that links virtual switch 1 in server R and virtual switch 2 in server G. Since the bandwidth of the physical link between the workstation and the FORE ATM switch is 155Mbps, the total bandwidth allocation for the VC's on the same physical links must be less than 155Mbps.

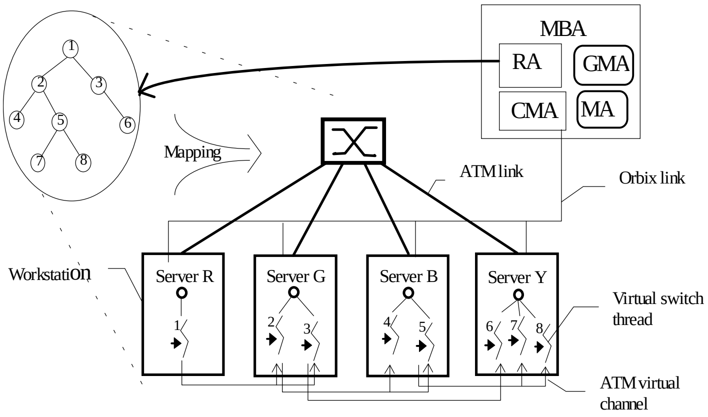

Figure 6. Simulation platform in CORBA environment

Orbix is a full implementation of the Common Object Request Broker Architecture (CORBA). It is a powerful environment for building and integrating distributed applications[2,6,8]. In Orbix, the components of a distributed program are objects - which have well-defined interfaces and may be communicated with from any node in the distributed system. In our simulation, an Orbix server manages a set of objects each simulating a virtual switch. Each server can have any number of clients communicating with these objects. As shown in Figure 6, the virtual switches are Orbix objects that are threads forked from the servers in a multi-threaded Orbix environment. The CMA in the Orbix client is responsible for establishing the multicast tree computed by RA. Whenever the CMA receives the topology of a multicast tree, it signals the Orbix servers in different workstations to initiate virtual switches and set up ATM links with reserved QoS between virtual switches according to the tree topology. After all the involved virtual switches are bound to the multicast tree by the ATM links, the CMA will signal the source to start data transmission. Another task of CMA is to add new links or tear down exist links that are computed by RA in response to any changes of group members.

3.2 Virtual switches

As articulated by many researchers, a revolution in network engineering may be achieved by replacing special-purpose switching hardware with a programmable, software-controlled switch. Such a switch consists of input and output ports controlled by a software-programmable element. This switches should be able to execute programs written in high level programming languages. We propose a virtual switch structure that simulates the logical functions of a programmable physical switch. This virtual switch, as depicted in Figure 7, consists of a VPI/VCI routing table, a control interface, a

Control and program Execution Unit (CEU), a program repository (e.g. filter bank) plus two signaling interfaces - one for input and the other for output.

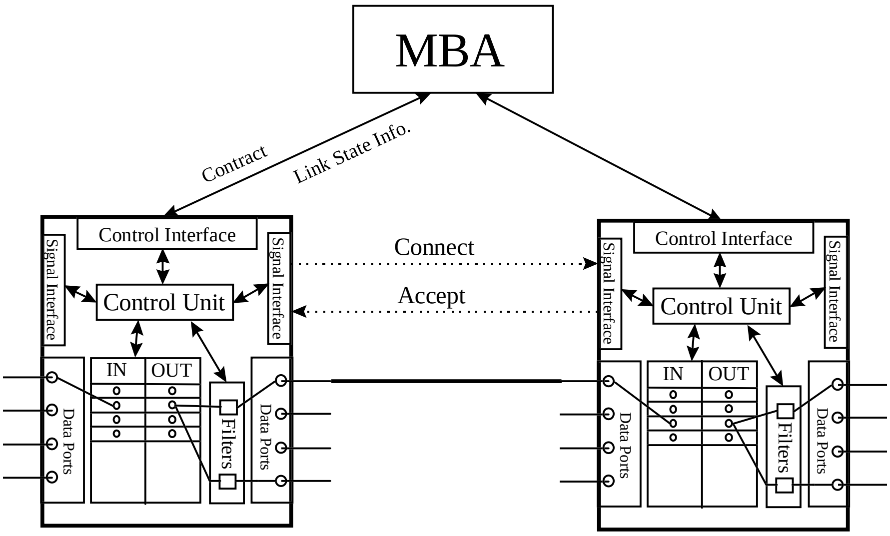

Figure 7. Structure and functions of the virtual switches

The control interface receives contracts from and sends link state information to MBA. To setup a multicast connection in a virtual switch, the VPI/VCIs must be reserved at both input port and output ports. Therefore, a virtual switch maintains the data abstraction of input and output VPI/VCI routing table that holds information about VPI/VCIs already assigned to the input and output ports. The CEU appends a new entry into the routing table whenever a contract is received from MBA. Normally, this entry connects to one incoming port and one or more outgoing ports. Moreover, each outgoing port may have an filter program inserted according to the injected contract. The received contracts are interpreted by the CEU that, in turn, signals the adjacent switches for establishing the required virtual channels through the signaling interfaces. The CEU first sets the specified incoming port to *waiting* state for receiving *connect* signal from the upstream switch. Whenever the *connect* signal is received from the upstream switch, the CEU checks the QoS request and sends out *accept* signal to the upstream switch if the request can be satisfied. The CEU sends out *connect* signals toward all the outgoing ports through the signaling interface and waits for ‘accept’ signals from all downstream switches. Subsequently, the filter programs are invoked from the filter bank and associated with the specified outgoing links. When all the links (i.e. incoming and outgoing links) are established and filter programs are in placed, the CEU sends back a link state message to signal that the contract is completed. In our current simulation platform, the VPI/VCI are emulated by the real ATM links using AAL5 protocol.

Table 1 lists some important control functions that are used to manipulate the routing tables and to setup or tear down the ATM connections.

Table 1. Description of the functions in a virtual switch

Methods	Functions
switchInitialization	Initialize the virtual switch and create and initializes the routing table.
setInMulticastVC	Set the incoming channel for a multicast session in the routing table and signal the upstream switch.
setOutMulticastVC	Set the outgoing channels for a multicast session in the routing table, signal the downstream switches, and set QoS parameters and associate filter for each channel.
deleteInMulticastVC	Delete an incoming channel from routing table, signal the upstream switch for leaving.

deleteOutMulticastVC	Delete output channels from routing table, signal downstream switches for breaking links.
addOneOutMulticastVC	Add one outgoing VPI/VCI in the routing table for a joining branch, signal the downstream switch for setting up a link.
deleteOneOutMulticastVC	Delete one outgoing VPI/VCI in the routing table for a leaving branch, signal the downstream switch for breaking down a link.
getInputChannelID	Get incoming port id and VPI/VCI.
getOutputChannelID	Get outgoing ports' id and VPI/VCI.

3.3 Simulation experiments

We used an N x N grid network as the primary simulation network topology in which each node connects to four neighboring nodes as shown in Figure 8. This topology provides a rich set of alternative paths between any two nodes and allows dynamically generate interesting random multicast trees. In our experiments, the simulation system transmits hierarchical encoded MPEG-1 streams to verified the different functions in the proposed active multicast service architecture. A MPEG-1 stream can be encoded into three sub-streams according to the types of the frames, namely I, P, and B streams[13,16]. Then the application displaying the MPEG-1 stream can require three levels of video quality by skipping some of the unnecessary frames. The basic level provides a lower level motion by carrying only the I stream. The second level carries I and P streams which provides a medium level motion performance by skipping the B-frames. The highest level service provides full motion video by carrying all the frames (I, P and B). In the simulation, an MPEG-1 stream of 30 Mbits is chosen. The receivers may select the desired video quality as high, medium and low which are corresponding to the number of frames 1211, 122 and 41 respectively. The amount of data transmitted for different video qualities and the required bandwidths are listed in Table 2. The video is originally designed to be played back at 25 fps and is 48.5 seconds in length. Therefore, to play back the video in real time, the minimum transmission rates are 606,842 bits/s, 198,477 bits/s and 84,963 bits/s for the high, medium and low video qualities respectively.

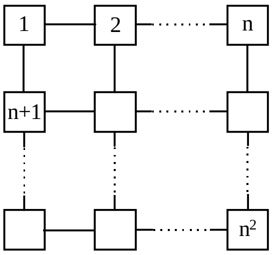

Figure 8. Network topology for simulation

Table 2. Required bandwidths for different video qualities

MPEG-1 stream	Low	Medium	High
Num. Of Frames	41	122	1211
Size of streams (bits)	4,120,712	9,626,144	29,431,816
Bandwidth required	128kbi/s	256kbits/s	640kbits/s

3.4 Simulation results

The experiments would ideally be performed on a large number of network nodes with variety of workload to ensure the robustness of the scheme. However, the sheer number of potential factors makes us to fix several network parameters. We have tested the scheme on the network with 9 to 64 virtual switches with various group sizes for multicast routing and resource reservation. The ‘artificial’ hierarchical coded MPEG-1 data is used to test the injected filter programs for selective data transmission. Current simulation results are promising in the sense that the system implements successfully the important concepts such as filter program invocation for selective data transmission, QoS reservation for heterogeneous receivers.

We simulated a multicast session to transmit the hierarchical encoded MPEG-1 stream with their features listed in Table 2. This session was performed in a network of 16 virtual switches and assumed that one workstation attached to each of the switches. As shown in Figure 9, the multicast source is the workstation connected to the switch 0 and the receivers are seven randomly selected workstations from the network. Each receiver can randomly select the required video quality of high, medium and low. The RA computes a multicast tree from the source to the seven receivers as shown in Figure 9. The reserved bandwidth for each data link is indicated in the figure as well. The filter programs are injected in the intermediate switches by MBA. For instance, there is a B-frame filter at Sw2 to drop B-frames for the outgoing link to Sw3. If all the receivers require low quality video, the total bandwidth reserved for this session is 1,480kbs. While if all the receivers select high quality video, the total bandwidth reserved for the session is 7,040kbs. This total bandwidth is the same as in normal multicast session without selective data transmission. However, if the receivers select quality of the video randomly as shown in Figure 9, the total bandwidth reserved for the session is 4,608kbs and this is about two thirds of the bandwidth required in a normal multicast session.

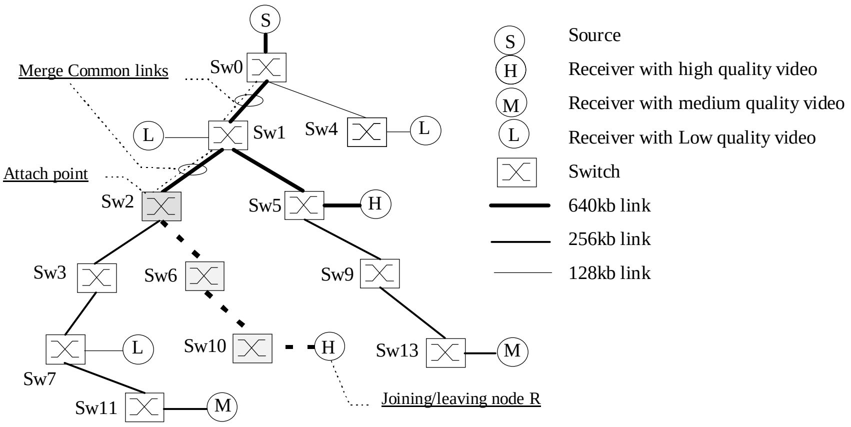

Figure 9. A multicast tree in a 16 nodes grid network with 7 group members

4. CONCLUSIONS AND DISCUSSIONS

This paper proposes an active multicast service architecture for ATM network. The architecture supports two levels of user customization of a multicast session. In the first level, users can select routing schemes by varying the weights of different link parameters in the distance function. In the second level, users can inject filter programs into the switches via a central management entity. This paper suggested some new functions to the ATM switches for efficiently supporting multicast applications. One function is that a switch has the capability to interpret the contracts that enables the switches to perform tasks specified by a centralized network management entity. Another function is the ability of the switch to execution user injected programs. Finally, the autonomous signaling mechanism for setting up virtual channels to neighboring switches implements an efficient parallel multicast tree setting up process. These switch functions can result in more powerful programmable switches.

The prototype system addressed important issues of the proposed Active Multicast Service Architecture. The prototype system provides functions that allow receiver initialized resource reservation for end-to-end QoS guarantee and filter program injection for selective data transmission of hierarchical encoded multimedia data. These features are achieved by using a centralized QoS negotiation and routing mechanism. Furthermore, the scheme supports nodes joining and leaving to a multicast group dynamically. The prototype system has been simulated in a simulation platform that is developed in a distributed CORBA environment. In this simulation platform, the functions of MBA are simulated by a CORBA client and the virtual switches are simulated as concurrent threads that are managed by CORBA servers. The links between virtual switches are emulated by AAL 5 virtual circuits to enable actual QoS reservation. The MBA communicates with virtual switches via the CORBA links that connect that MBA and CORBA daemons residing in virtual switches.

From the simulation results, we can conclude that supporting active technology inside the network gives users more flexibility in customizing their applications. Experimental results also show that this multicast architecture provides a greater flexibility in satisfying the users of different capabilities and QOS requirements. In addition, by fine tuning resource allocation among heterogeneous users can save the total resource (e.g., bandwidth) required for a multicast session while provides reasonable quality of data transmission.

We believe that networks with enabling active technologies will eventually lead to a user-driven innovation process, in which the availability of new services will not be delayed by vendor consensus and standardization activities. There are at least two potential problems need to be further investigated before the proposed services can be deployed in the actual networks that spread over large geographical areas. Both problems are related to the centralised MBA approach: scalability and reliability of MBA in large networks. A possible solution is to segment the large network into smaller clusters each of which is managed by a MBA. This may address some of the potential setbacks of centralized multicast management mentioned earlier on, but also raise the federation problems among different MBAs which has to be investigated further. Another consideration is what kind of common open control and signaling interfaces be implemented eventually.

ACKNOWLEDGEMENT

This work was supported by a National University of Singapore research grant RP940608 and a National Science and Technology Board post-doctoral research grant.

REFERENCES

1. K. G. Murty, *Network Programming*, Prentice Hall, 1992.
2. IONA Technologies Ltd. *Orbix Advanced Programmer's Guide*, July 1995.
3. A. Alles, *ATM Internetworking*, Cisco Systems, Inc., Mar. 1995.
4. ATM Forum, *ATM User-Network Interface Specification Version 3.0*, Prentice Hall, 1993.
5. G. Armitage, "Support for Multicast over UNI 3.1 Based ATM Networks," *Internet Draft*, Jan. 1995.
6. M. C. Chan, J. F. Huard, A. A. Lazar and K. Lim, "Service Creation, Renegotiation and adaptive Transport for multimedia Networking," to be published in the *Third COST 237 Workshop on Multimedia Telecommunications and Applications*, Barcelona Spain, November 25-27, 1996.
7. N. Shacham, "Multipoint Communication by Hierarchically Encoded Data," *IEEE Proceedings of INFOCOM '92*, 1992, pp. 2107-2114.
8. Object Management Group(OMG), *Common Object Request Broker: Architecture and Specification*, Document 91-12-1.
9. A. Compbell, G. Coulson, D. Hutchison, "A Quality of Service Architecture," *ACM Computer Communications Review*, Vol. 24, No.2, pp. 6-27, 1994.
10. L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A New Resource ReSerVation Protocol," IEEE Network, Sept. 1993.
11. R. Braden, L. Zhang, D. Estrin, S. Herzog, S. Jamin, "Resource Reservation Protocol -- Version 1 functional Specification," *Internet Draft*, 1995.
12. L. H. Ngoh, H. Y. Li, H. K. Pung, "A Direct ATM Multicast Service with QoS Guarantees," in Proceeding of IEEE Conf. On Multimedia'96, Japan, pp. ##, July, 1996.
13. J. K. Buford, "Multimedia Systems," Addison Wesley, 1994.
14. R. Widyono, "The Design and Evaluation of Routing Algorithms for Real-time Channels," *Tenet Group TR-94-024*, U.C. Berkeley, 1994.
15. D. L. Tennenhouse, D.J.Wetherall, "Towards an Active Network Architecture," to appear in *Computer Communication Review*, 1996. Http://www.tns.lcs.mit.edu/publications/ccr96.html.
16. N. Yeadon, F. Garcia, D. Hutchison, D. Shepherd, "Continuous Media Filters for Heterogeneous Internetworking," *IS&T/SPIE International Symposium. on Electronic Imaging Special Session on Multimedia Computing and Networking*, San Jose, California, Jan. 1996.
17. T.E. Bell, J. A. Adam, S.J. Lowe, "Technology 1996 - Communications," *IEEE Spectrum*, Jan 1996, pp. 30-41.
18. J. M. Smith, "SwitchWare: Accelerating Network Evolution with a Software switch for Active Network," http://www.cis.upenn.edu/~jms/SoftSwitch.html